TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JORDI MORAL

STUDY AND DEVELOPMENT OF A KANBAN TOOL

Master of Science Thesis

Examiner: Professor Kari Systä
Examiner and topic approved by the
Council of the Faculty of Computing
and Electrical Engineering on
4 June 2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
MORAL DALMAU, JORDI: Study and development of a Kanban tool.
Master of Science Thesis, 47 pages, 6 Appendix pages
June 2014
Major: Software Engineering
Examiner: Kari Systä
Keywords: Agile Software Development, Kanban, Scrum, Open Source, Project Management Tool.

The use of Agile software development has increased in a significant way the recent years. The Agile methodologies can be much more efficient in comparison than classic methods. One of the most useful Agile practices is Kanban.

The main goal of this thesis is to go deep into the Kanban practices, finding out and understanding the Kanban capabilities that can provide a better project management. Apart from the study, an Open Source web oriented Kanban tool has been developed to understand Kanban in a closer way. The tool is called KanBoard and it can be very useful towards students and small development teams.

The thesis is divided into two parts. In the bibliographic part, Agile software methodologies are introduced, studied, compared and evaluated. In the other part, the tool implemented is detailed together with the purpose, the features and its evaluation.

RESUM

TAMPERE UNIVERSITY OF TECHNOLOGY
Grau en Enginyeria Informàtica
MORAL DALMAU, JORDI: Estudi i desenvolupament d'una eina Kanban.
Treball Final de Grau, 47 pàgines, 6 pàgines d'apèndix
Juny 2014
Menció: Enginyeria del Software
Examinador: Kari Systä
Paraules clau: Desenvolupament Àgil del Programari, Kanban, Scrum, Codi Obert, Gestió de Projectes.

L'ús dels mètodes Àgils en el desenvolupament del progrmari s'ha incrementat d'una manera significativa en els darrers anys. Les metodologies Àgils poden ser molt més eficients en comparació amb els mètodes clàssics. Un dels mètodes Àgils més utilitzats actualment és Kanban.

L'objectiu principal d'aquesta tesi és aprofundir en el mètode Kanban i estudiar quines són les pràctiques que permeten una millor gestió dels projectes. Per altre banda, s'ha desenvolupat una eina que permet entendre el Kanban d'una manera més clara, senzilla i propera. Aquesta eina s'ha anomenat KanBoard i un dels seus objectius pels quals ha estat dissenyada és perquè pot ser útil pels estudiants i els petits grups de desenvolupament de software.

La tesi es divideix en dues parts. A la part bibliogràfica, s'introdueixen els diferents mètodes Àgils, seguidament d'un estudi, una comparació i una avaluació dels més importants. A l'altre part, es detalla com s'ha implementat l'eina, juntament amb els seus objectius, característiques i avaluació.

PREFACE

With this thesis I conclude a stage at university life and also an experience in a foreign country, Finland. This project was proposed by the professor Kari Systä. Thanks to this work, I realized that Agile can be a useful way in the future software development.

I would like to thank Kari Systä for giving me the opportunity to work on this project and for all the support and encouragement received during all this period. I would also like to thank all the people that helped me spending their own time testing the tool and writing their feedbacks.

I am grateful to Alfonso Cambero for contributing his expertise in the development process and Enric Omaña as a language consultant. And last but not least I would like to thank my girlfriend, Carla Cordomi, fellow friends and parents for their moral support.

I hope you will enjoy reading this thesis.

JORDI MORAL
June 2014

TABLE OF CONTENTS

TERMS AND DEFINITIONS

CSS	Cascading Style Sheets
DOM	Document Object Model
FTP	File Transfer Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
LAMP	Software bundle composed by Open Source software
MMF	Minimum Marketable Feature.
MVC	Model-View-Controller architecture pattern
OS	Operating System
PHP	Hipertext Preprocessor
SLA	Service-Level Agreement
SQL	Structured Query Language
SSH	Secure Shell
UI	User Interface
UX	User Experience
WIP	Work In Progress
XML	Extensible Markup Language

1. INTRODUCTION

1.1 The Topic

From the beginning I wanted to develop a useful project management tool, then, my professor suggested me the Kanban practices. After a few days reading about the topic I realized how powerful the Kanban was and the more I researched about it the more I liked it, so I was quickly interested to go deeper. Then I decided to introduce me into the new paradigm of the Agile development. How Kanban use is increasing in Agile software development teams and seems to will increase in the near future, I though that developing a tool to satisfy the previous arguments could be a very suitable task.

1.2 Purpose

The purpose of this thesis is to develop a project management tool to support Agile software development that could be useful in both, professional and personal projects. The tool main target consists in digitalizing the physics Kanban boards into a browser application to improve the interaction and collaboration of a team. With this project, there has been a small contribution to the Agile software development.

1.3 Structure

This project is composed by two parts. In the bibliographic part, a short study of Agile software development and a comparison of the most used methods is given. In the other part is written all the information related with the tool developed. In this last part are included the features, the implementation and a short demonstration of the tool. There is also an evaluation with some feedbacks and future improvements. Finally, in the conclusions, a final project assessment is achieved.

2. AGILE SOFTWARE DEVELOPMENT

Software development has changed over the time, and the Agile software development emerged as a part of a reaction against the well-structured and strict methods that came from the waterfall-oriented methods, which were characterized by their critics as being heavily regulated, regimented and overly incremental approaches to development.

The waterfall process has been seen as bureaucratic, slow, degrading and inconsistent model with the ways of software development that performed an effective job because of the poorly adaptable to changing requirements and the overhead to be balanced between the time spent planning and the time that fixing a defect would actually cost.

The use of the word Agile in this context derives from the Agile Manifesto. A small group of people got together to discuss their feelings that the traditional approach to managing software development projects was failing far too often, and there had to be a better way. The Agile Manifesto was written in February of 2001 in Snowbird, Utah resort, at a summit of seventeen independent minded practitioners of several programming methodologies, where they met to discuss lightweight development methods. The participants didn't agree about much, but they found consensus around four main values that nowadays sustains the fundamentals of Agile development.

Values proclaimed by the Agile Manifesto [1] were:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

Those values can be explained such that in Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming. Working software will be more useful and welcome than just presenting documents to clients in meetings. The requirements cannot be fully collected at the beginning of the software development cycle, therefore continuous customer or stakeholder involvement is very important. Agile development is focused on quick responses to change and continuous development.

The whole purpose of the Agile Manifesto is to deliver better software. Is more needed to focus on value and results than add to a firm's competitive advantage.

Principles of Agile Manifesto [1]:

- *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*
- *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
- *Business people and developers must work together daily throughout the project.*
- *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*
- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working software is the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
- *Continuous attention to technical excellence and good design enhances agility.*
- *Simplicity--the art of maximizing the amount of work not done--is essential.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.*

Some of the previous principles are related to the regular delivery of software. Software that is the true deliverable is one of the last things to be delivered to the customer. In contrast, Agile principles focus on the software, which is the most important thing, being delivered early and often. It is common knowledge that teams need to communicate in order to be successful, so other principles emphasize the team communication. To maximize an Agile approach, is needed to design and build excellent software solutions. Some principles emphasize these points and they allow accommodating change. Like most of Agile, it is a journey that requires vigilance and attention to changing events.

We can say that Agile is not an entity, not a framework, neither a methodology; it is a collection of values and principles that encourages a certain type of behaviour. It is based on the iterative and incremental development, where the requirements and solutions evolve trough the collaboration of a self-organized and cross-functional teamwork.

The majority of the Agile methods are based in a short iterations, and the tasks are break into small increments. All iteration involves planning, requirements analysis, design, coding, unit testing and acceptance testing. At the end of the iteration, a working product is demonstrated to stakeholders. This minimizes overall risk and allows the project to adapt to changes quickly. At the end of each iteration, stakeholders and custom-

ers representative, review progress and re-evaluate the priorities and ensure the alignment between customer needs and company goals.

In Agile software development, the working team is located at the same physical place, normally in an office, face-to-face communication is used instead the large documentation, and a board to inform a team about the current status of their project. Specific techniques are used in Agile development to enhance project agility such as continuous integration, automated unit testing, pair programming, test-driven development, design patterns, domain-driven design, code refactoring, etc.

From the Agile Manifesto, Agile practises have increased considerably, even so, have been strongly criticized and treated as an undisciplined because of the absence of technic documentation.

Some of the most used agile practices are Scrum, eXtreme Programming (XP), Kanban and Crystal Clear. Despite that, some companies use hybrids between them to get the best from some of them.

Nowadays, some companies are unwilling to adopt Agile methodologies due to they think that they are not mature enough. Others use them only in some particular projects, and some others have changed entirely the developing way to Agile.

Is not adequate to say the Agile methods are the best way to develop software putting aside the traditional methods, each company have to adopt the process that fits them better.

Lean Software Development

Lean principles originate from the lean manufacturing approach also known as ‘just-in-time production’, and pioneered by Toyota. Lean Manufacturing is a process management philosophy that transformed the car manufacturer’s approach to building vehicles. A set of principles and behaviours related to managerial approach and production system are described in a book called The Toyota Way.

Lean Software Development is an Agile practice that is based on the principles of Lean Manufacturing. Lean Software Development comes from the book "Lean Software Development: An Agile Toolkit" by Mary and Tom Poppendieck published in 2003 and is based on 7 principles and 22 tools detailed in the book.

Lean Software Development is not a management or development methodology per se, but it offers principles that are applicable in any environment to improve software development.

The 7 Principles of Lean Software Development:

- **Eliminate waste.** Lean thinking advocates regard any activity that does not directly add value to the finished product as waste. The three biggest sources of waste in software development are the addition of unrequired features, project churn and crossing organizational boundaries. To reduce waste it is critical that

development teams be allowed to self organize and operate in a manner that reflects the work they're trying to accomplish.

- **Build in quality.** The process should not allow defects to occur in the first place, but when this is not possible, the defect should be worked, validated, fixed and iterated. Inspecting after the fact, and queuing up defects to be fixed at some time in the future, isn't as effective.

- **Create knowledge.** Planning is useful, but learning is essential. It is important to promote strategies, such as iterative development, that help teams discover what stakeholders really want and act on that knowledge. And for a team, is necessary to regularly reflect on what they are doing to improve their approach.

- **Defer commitment.** It is not necessary to start software development by defining a complete specification. The business can be supported effectively through flexible architectures that are change tolerant and by scheduling irreversible decisions to the last possible moment.

- **Deliver quickly.** It is possible to deliver high-quality systems quickly limiting the work of a team to its capacity, which is reflected by the team's velocity. Constraining the teams to delivering potentially shippable solutions on a regular basis motivates them to stay focused on continuously adding value.

- **Respect people.** A sustainable advantage is gained from engaged, thinking people. Is needed a lean governance strategy that focuses on motivating and enabling IT teams, not on controlling them.

- **Optimize the whole.** To be effective at a solution should be looked at the bigger picture. Is needed to understand the high-level business processes that individual projects support. Is needed to manage programs of interrelated systems to deliver a complete product to stakeholders. Measurements should address how well you're delivering business value, because that is the sole reason for your IT department.

Agile Adoption in the Enterprise

According to a survey made in 2013 by VersionOne Inc., from a broad range of organizations sizes, varying from single-employee businesses to those with more than a hundred thousand. ¾ were from smaller organizations (fewer than 1,000 employees).

It can be clearly seen that the most popular Agile method used is Scrum and Scrum variants (73%). Kanban and Scrumban cover only the 12% of usage, but comparing with two years ago has increased in 50%.

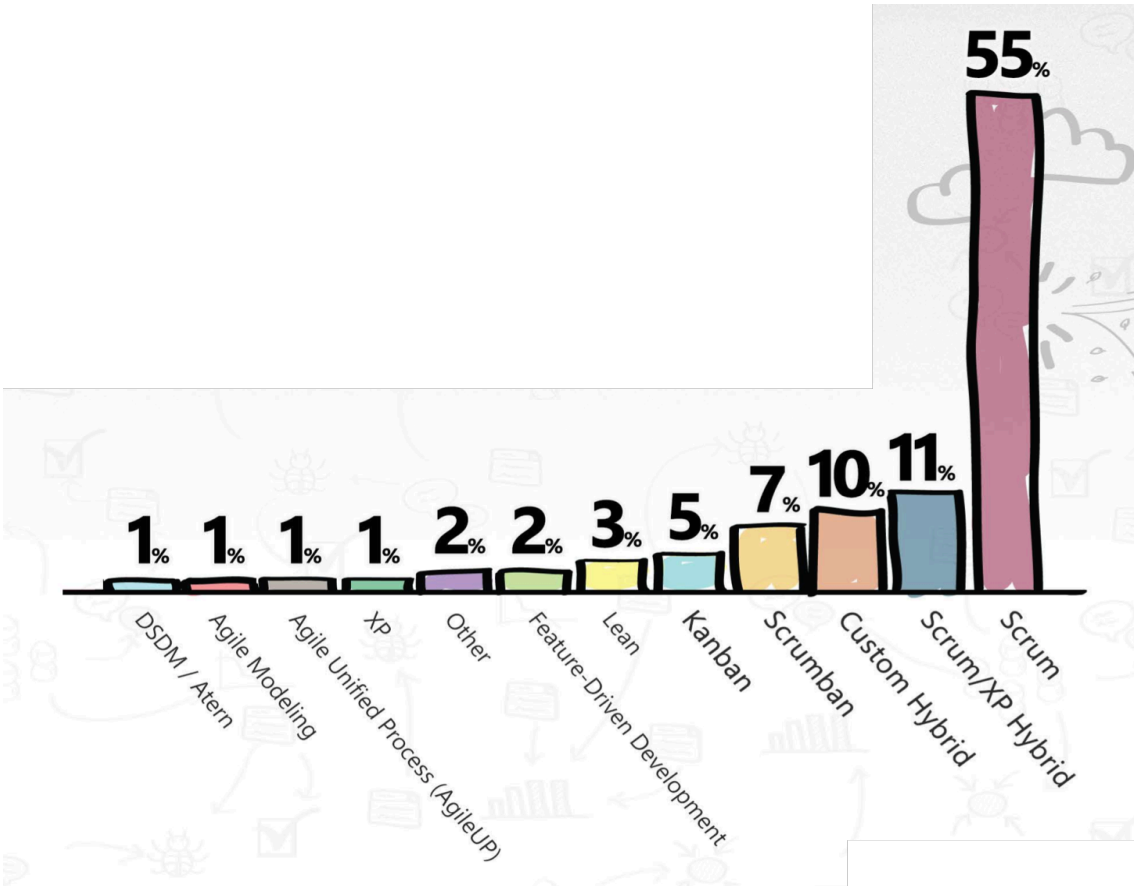


Figure 2.1. *Percentage of the use of agile methodology. [4]*

Over half of the surveyed companies manage between 50% and 100% of their projects using Agile. Nearly 40% are managing ¾ to all of their projects using Agile. The number of teams practicing Agile is continuously increasing, this previous year the 57% of the companies had adopted Agile practices across 5 or more teams. In addition to scaling agile across more teams, organizations are also scaling Agile to a greater number of projects.

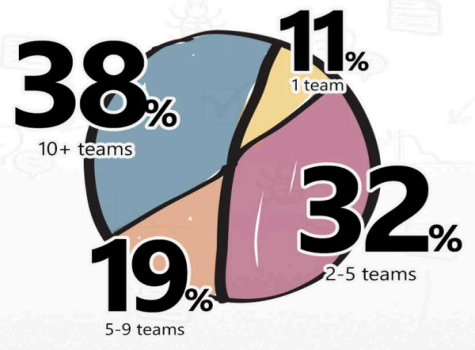


Figure 2.2. *Agile adoption on teams. [4]*

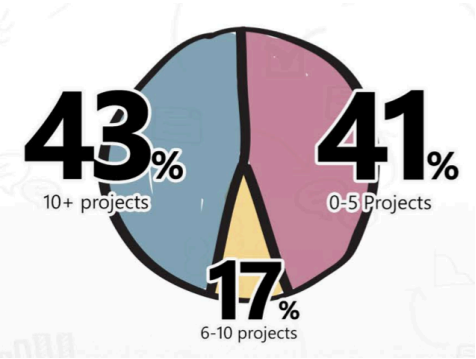


Figure 2.3. *Number of projects using agile. [4]*

About ½ the respondants who use Kanban or Scrumban said they were primarily using these methods for business processes inside the software organization only.

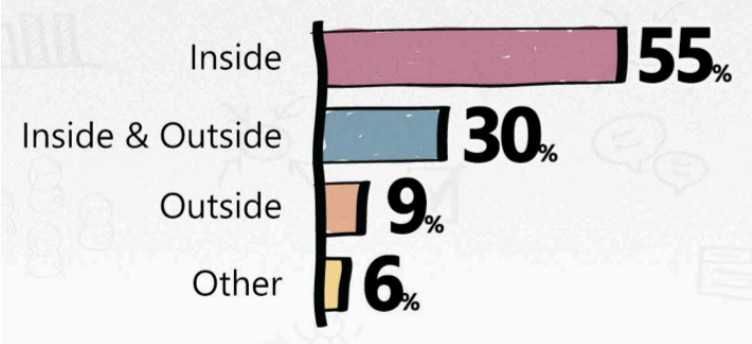


Figure 2.4. *Percentage of how Kanban is applied. [4]*

2.1 Scrum & Kanban

Basically **Scrum** can be defined in three sentences:

- Split the organization into small, cross-functional, self-organized teams.
- Split the work into a list of small, concrete deliverables, sorting the list by priority and estimating the relative effort of each item.
- Split the time into short fixed-length iterations, about 1-4 weeks, with shippable code demonstrated after each iteration and optimize the process by having a retrospective.

On the other hand **Kanban** has a higher degree of freedom in terms of the rules to be applied and the basics are:

- Create a board with some states, split the work into pieces, write each item on a card and put on the wall.
- Limit Work in Progress (WIP); assign explicit limits to how many items may be in progress at each workflow state.
- Measure the lead time, the average time to complete one item, and optimize the process to decrease it.

Neither method tell everything needed to success, they just provide certain constraints and guidelines. Scrum constrains to have timeboxed iterations and cross-functional teams, and Kanban constrains to use visible boards and limiting the size of the queues.

2.1.1 Prescriptive vs Adaptive

Scrum and Kanban are lightweight methods that are less prescriptive than traditional methods, if we understand prescriptive as more rules and adaptive as less rules to follow. Even so, Scrum is more prescriptive than Kanban.

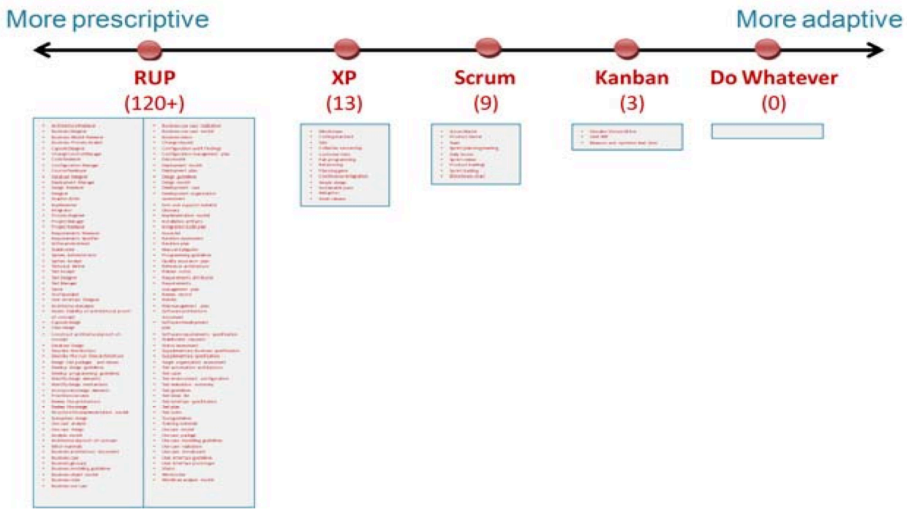


Figure 2.5. *Comparison of process tools on the prescriptive vs adaptive scale. [8]*

2.1.2 Lean and Agile

Scrum and Kanban are both pull scheduling systems, the team chooses when and how much work they want to commit. Scrum and Kanban are based on continuous and empirical process optimization, which corresponds to the Kaizen (continuous improvement aims to eliminate waste) principle of Lean. Both Kanban and Scrum emphasize responding to change (one of the four values for Agile Manifesto) over following a plan, although Kanban typically allows faster response than Scrum.

2.1.3 Teams

In Scrum, the sprint backlog shows what tasks need to be executed during the current iteration and is commonly represented using cards on the wall, called a Scrum board. The Scrum task board is divided by columns representing the states where the tasks cross.

A Scrum board is owned by exactly one Scrum team. Scrum Teams are self-organizing and cross-functional. Self-organizing teams choose how best to accomplish their work, rather than being directed by others outside the team. Cross-functional teams have all competencies needed to accomplish the work without depending on others not part of the team.

In Kanban, cross-functional teams are optional, and a board does not need to be owned by one specific team. A board is related to one workflow, not necessary one team.

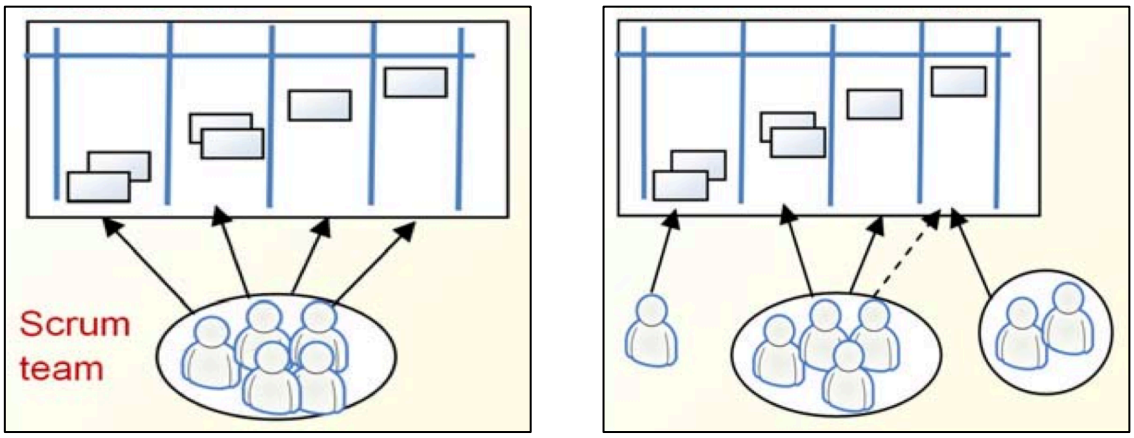


Figure 2.6. *On the left, example of a cross-functional Scrum team. On the right, example of different tems working on the same Kanban board. [8]*

2.1.4 Roles

In Scrum 3 roles are prescribed: Product Owner, Scrum Master and the Team. The Product Owner sets the product vision and the priorities, the Scrum Master removes impediments and provides process leadership and the Team is who implements the product. Kanban does not prescribe any role at all, but that does not mean that cannot incorporate the Product Owner role.

In Scrum, prioritization is always done by the Product Owner, who sorts the product backlog and changes the priorities that take effect in the next sprint. In Kanban, the prioritization scheme is not prescribed and changes take effect as soon as capacity becomes available. The Product Owner cannot touch the Scrum board since the team has committed to a specific set of items in the iteration, only is allowed to edit the product backlog. In Kanban, the team set their own ground rules for who is allowed to change what on the board.

2.1.5 Meetings

When the sprint is over, the Scrum board is cleared and all items are removed. A new sprint is started and after the sprint planning meeting there is a new Scrum board, with new items in the left-most column. In Kanban, the board is normally a persistent thing, is not needed to reset and start over.

Scrum team has a short meeting every day at the same time and in the same place called *daily standup*. The purpose of the meeting is to spread information about what is going on, plan the current day's work, and identify any significant problems. *Daily standups* are not prescribed in Kanban, but most teams seem to do it anyway.

In Scrum the format of the meeting is people-oriented, every person reports one by one. Many Kanban teams use a more board-oriented format, focusing on bottlenecks and other visible problems.

2.1.6 WIP per Iteration vs WIP per Workflow

Scrum is based on timeboxed iterations; the idea is to keep the same length of iteration over a period of time. In the beginning, an iteration plan is created and the team pulls out specific number items from the product backlog. During the iteration the team focuses the items they are committed to. At the end the team demonstrates working code to stakeholders and they do a retrospective to discuss and improve their process. So, Scrum iteration is one single timeboxed cadence combining planning, process improvement and release. In Kanban, timeboxed iterations are not prescribed, each team choose to do their activities on a regular basis (release every week) or on-demand (release whenever something useful to release).

The difference between the Scrum board and a Kanban board is the visualization of WIP (Work in Progress). Let's see a simple example:

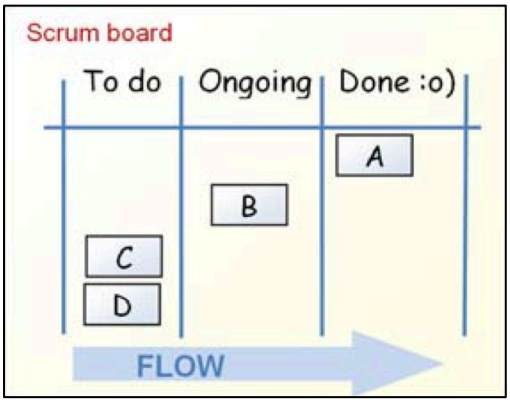


Figure 2.7. *Simple example of a Scrum board. [8]*

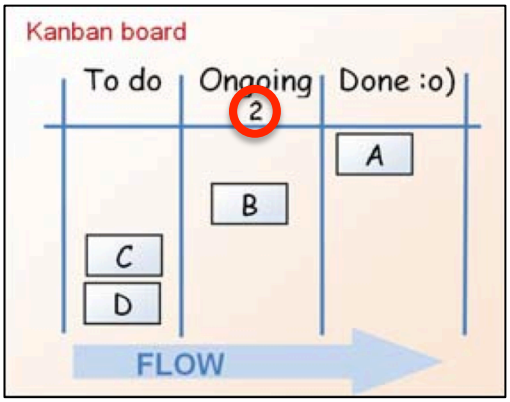


Figure 2.8. *Simple example of a Kanban board. [8]*

Both cases are tracking a bunch of items as the progress through a workflow. Are selected three states: *To Do*, *Ongoing*, and *Done*. Each team choose the states that they prefer, some of them add states such as *Integrate*, *Test*, *Release*, etc. The difference between these two sample boards is the 2 in the middle column on the Kanban board. And that 2 means there may be no more than 2 items in this column at any given moment.

In Scrum there is no rule preventing to put all items into one state at the same time, however, there is an implicit limit since the iteration itself has a fixed scope. In this case the implicit limit per column is 4, since there are only 4 items on the whole board. So Scrum limits WIP indirectly, while Kanban limits WIP directly.

So both Scrum and Kanban limit WIP, but in different ways. Scrum teams usually measure velocity, how many items get done per iteration. Once the team knows their velocity, that becomes their WIP limit.

In Scrum WIP is limited per unit of time and in Kanban WIP is limited per workflow state. In Kanban the general idea is to limit WIP of all workflow states and then, start measuring and predicting lead time. Having predictable lead times allows committing to SLAs and making realistic release plans.

2.1.7 Estimation and Velocity

Both Scrum and Kanban are based on incremental development, which is breaking the work into smaller pieces. A Scrum team only commit to items that they think they can complete within one iteration.

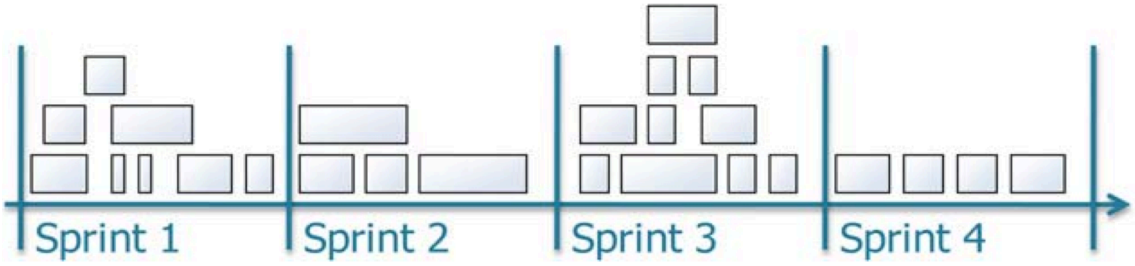


Figure 2.9. *Example of the duration of different items to be done in every sprint. [8]*

Kanban teams try to minimize lead time and level the flow, but there is no explicit rule stating that items must be small enough to fit into a specific time box. On the same board there might have one item that takes one month to complete and another item that takes one day.

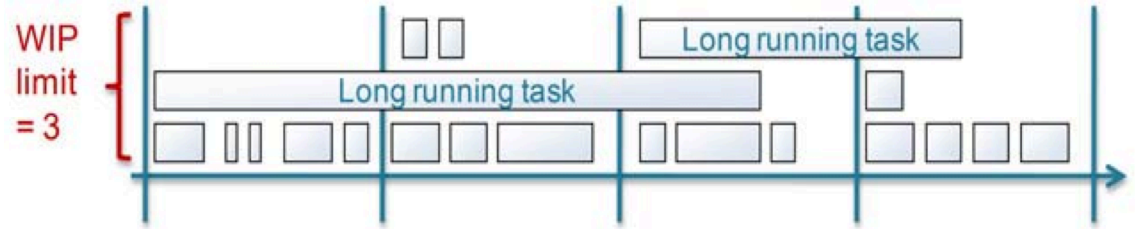


Figure 2.10. *Example of the duration of different items in Kanban. [8]*

In Scrum, the teams are supposed to estimate the relative size of each item that they commit to, adding up the size of each item completed at the end of each sprint. This is known as velocity and measures how much stuff can be delivered per sprint.

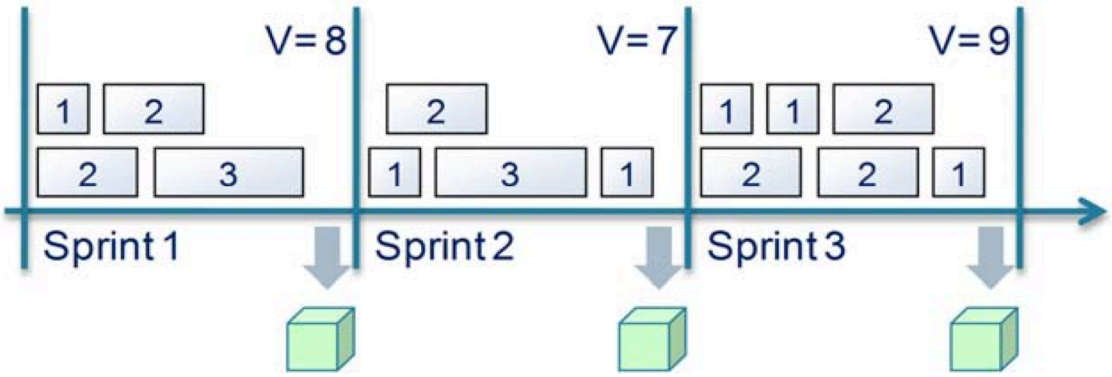


Figure 2.11. *Example of the velocity on every sprint in Scrum. [8]*

In Kanban, the estimation is not prescribed. Some teams may chose to make estimates and measure velocity just like in Scrum, other try to break each item into pieces of roughly the same size and some group items into MMFs (Minimum Marketable Feature) and measure the average lead time per MMF. There is all kinds of interesting techniques for Kanban-style release planning and commitment management.

2.1.8 Continuous Improvement

Scrum and Kanban are both empirical meaning each team is expected to experiment with the process and customize to their environment, team draw conclusions from the results and then change some more things, continuously improving the process. For these previous processes there are many names; Kaizen (continuous improvement in Lean-speak), Inspect & Adapt (Scrum-speak), Empirical Process Control…

In Kanban the basic feedback comes from the average lead time and bottlenecks. In order to avoid bottlenecks and reduce the lead time, adjusting the WIP value in the states will reduce idle times and maintain a continuous flow without bottlenecks. If the WIP limit is too low, there will be idle people and finally a bad productivity, if the WIP limit is too high, there will be idle tasks and a bad lead time.

2.1.9 Charts

In Scrum, sprint burndown charts are used as one of the primary tools for tracking the progress of iteration. A burndown chart shows, on a daily basis, how much work remains in the current iteration and the main purpose is to easily find out as early as possible if the team is following the schedule. In Kanban, no particular type of chart is prescribed.

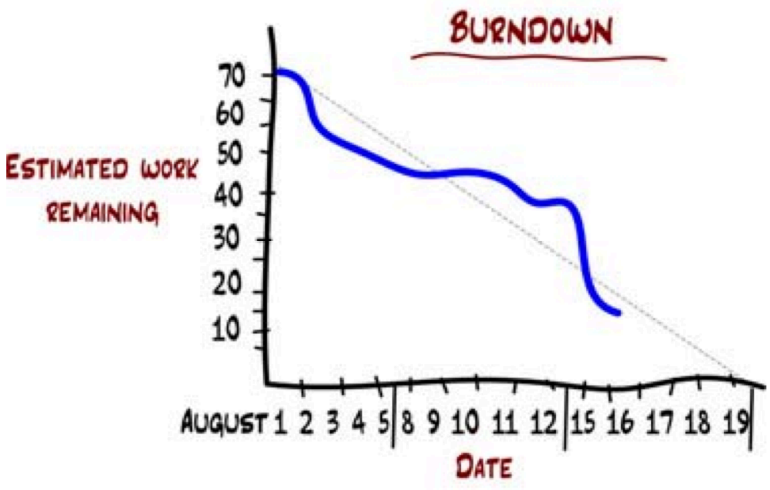


Figure 2.12. *Example of a burndown chart. [8]*

Another kind of chart commonly used in Kanban is the Cumulative Flow diagram. This type of chart illustrates nicely how smooth the flow and how WIP affects the lead time. The down and across arrows illustrate the relationship between WIP and lead time.

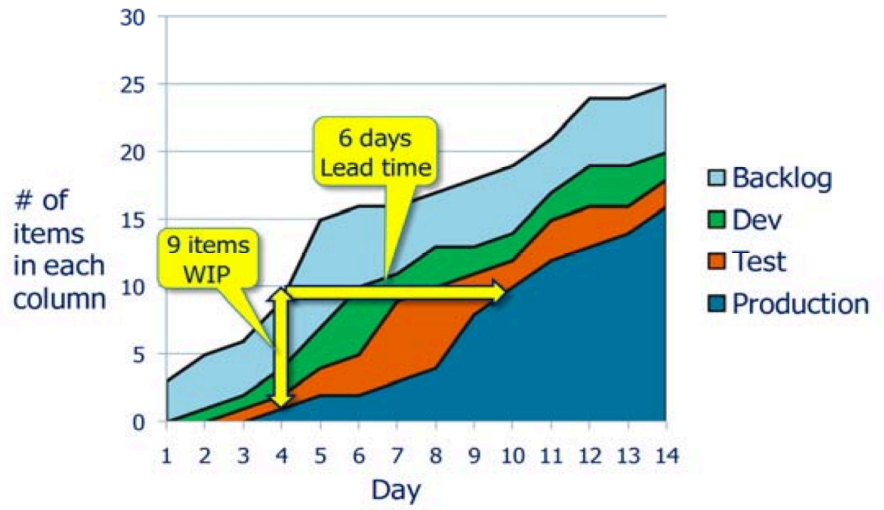


Figure 2.13. *Example of a Cumulative Flow diagram. [8]*

If we look to the previous example, we can see that the horizontal arrow shows us that items added to the backlog on day 4 took on average 6 days to reach production. About half of that time was Test. We can see that if were to limit the WIP in Test and Backlog we would significantly reduce the total lead time. The slope of the dark-blue area shows us the velocity and over the time we can see how higher velocity reduces lead time, while higher WIP increases lead time.

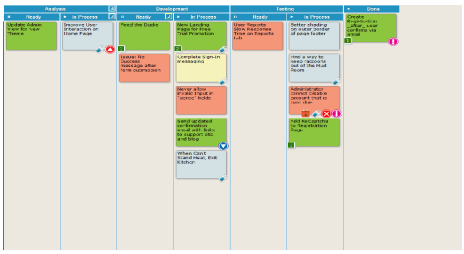
2.2 Evaluation of existing tools

Nowadays, there are many different Kanban tools in the market. A brief list of some tools that have featured as the most useful hosted oriented Kanban tools are listed and separed by commercial tools, free to use tools (with the possibility to pay for more features) and Open Source tools.

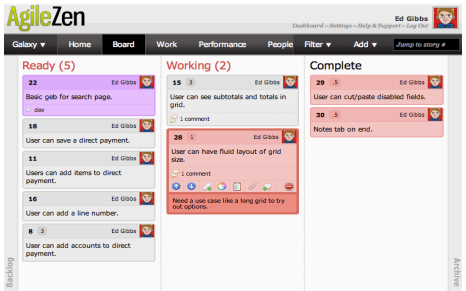
Commercial Kanban Board Tools

- KanbanTool – It is one of the most famous online tools that cover all Kanban needs. It is pretty simple to use thanks to the coloured notes and the drag & drop interface. It has a real-time response, control and optimization the workflow with WIP limits and multiple projects in the same board. This tool also provides a good analytics and monitoring with different charts and the possibility to upload and share documents.
  It has a free plan limited to two collaborators and two boars.
  http://kanbantool.com

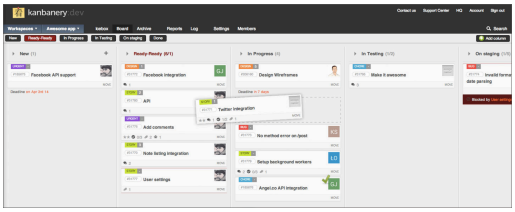
- LeanKit – It is a complete tool collaborative in real time with plenty of metrics and features that provides a simple or complex workflow creating sub-lanes to represent different teams. It also provides real meaningful metrics that shows where to focus more efforts, if someone is overloaded and how effective is the process.
  There is a free plan up to ten collaborators limited to three boards.
  http://leankit.com



- AgileZen – It is a really simple tool with a very easy-to-use web interface. Each project has a backlog of upcoming stories and an archive of old stories. The cards have different colors to separate the scope. Provides reports for lead time, cycle time, work time and wait time. It is not possible to attach files.
  The free plan is limited to one collaborator and one board.
  http://www.agilezen.com

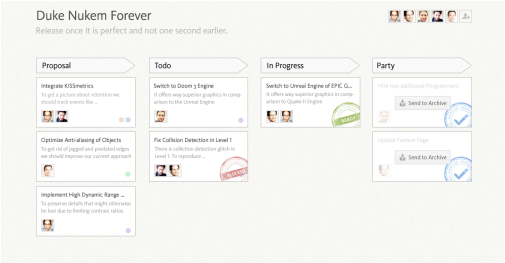


- Kanbanery – Kanbanery is a really well designed Kanban tool, with a pretty user interface. It offers simple team collaboration and communication tools, file sharing, real-time updates and advanced reporting. Provides GitHub integration.



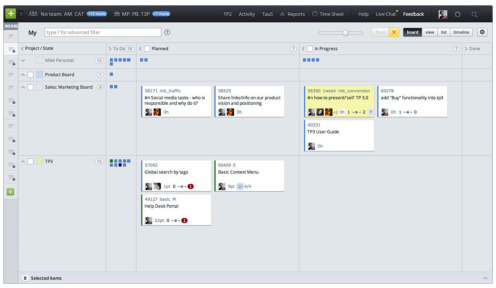
  Free plan limited to two collaborators and two users.
  https://kanbanery.com

- Blossom – It has the simplest and minimal interface that provides the Kanban basics. It is really simple to use and includes some features such as checklists, email notifications and digests, GitHub integration and a cycle time performance analytics to identify outliers.



  There is not any free plan, only a free trial of fourteen days.
  https://www.blossom.io

- TargetProcess – It is a very powerful tool with integration of releases, sprints, inbuilt version control, defect management, helpdesk systems and collaborative mechanisms. It provides a lot of features to customize the board and the cards, lists and timeline visualizations and mobile access.

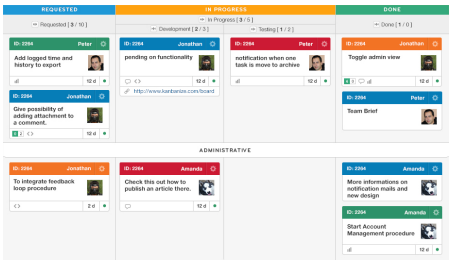


  It is completely free up to five collaborators.
  http://www.targetprocess.com

Apart from the previous ones, there are many tools that are also very interesting such as Jira, Axosoft, Hansoft, AgileWrap, Swift-Kanban, VersionOne or Yodiz.

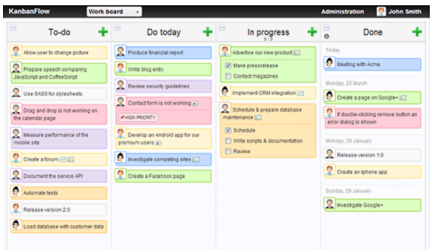
Free Kanban Tools

- Kanbanize – Kanbanize is a very useful tool, with unlimited collaborators and boards. It Allows teams manage the constant workflow in real time. It is very simple but at the same time it has plenty of features such as customizable boards and cards, user privileges depending the role, a powerful analytics module provided by different charts, reporting and monitoring and mobile support.



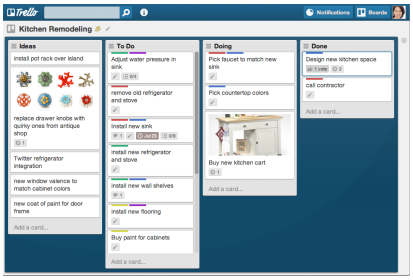
  Completely free, also payment plans with more features.
  https://kanbanize.com

- KanbanFlow – This is a good option for a simple Kanban needs, it provides collaboration in real time, subtasks in each item, file attaching, some charts to analyse and evolve the process and mobile support.



  Is free to charge for unlimited boards and collaborators, with option to pay for premium features.
  https://kanbanflow.com

- Trello – This is not a complete Kanban tool, and it is more oriented to Scrum needs. Although is a remarkable and very popular free option, doesn't support WIP limits. But to solve that, there is an extension for Google Chrome that allows implementing WIP. Provides almost all the features of the best tools, except the analytics charts.

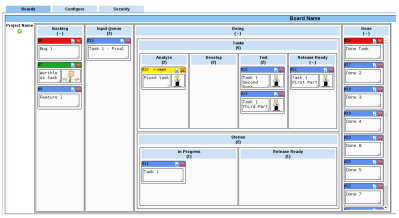


  Its free to use, also with bussines and gold payment options.
  https://trello.com

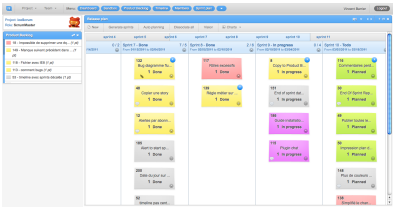
Other options less remarkable are Kanban2go and KanbanPad.

Open Source Tools

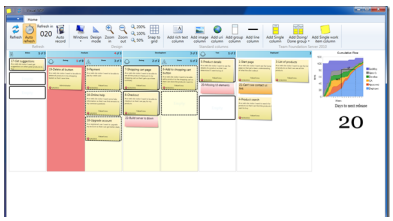
- Kanbanik – It is a free and Open Source Kanban board that can be used for personal use or managing small teams. It is the most powerful Open Source tool at the present time. Supports complex flows with sub-lanes, drag and drop option for the tasks and possibility to assign tasks to a user. It is a web application optimized for Google Chrome and exists a runtime to install the server on Windows and Linux machines.
  https://code.google.com/p/kanbanik/



- IceScrum – It is a tool focused in Scrum development but also implements some Kanban practices like WIP limits. The boards can be shared between teams and it also provides indicators as charts. There is available a war file to download or to install a server on Windows or Mac.
  http://www.icescrum.org



- VisualWIP – It is an Open Source desktop application, which provides a very customizable board and some parameters to set. VisualWIP also shows a cumulative flow chart to monitor and analyze the process.
  http://visualwip.codeplex.com



There are also some Open Source projects being developed but they are not finished or they have been abandoned such as:

- Kanban: an Open Source Trello clone
  http://softworkr.com/blog/20140206-open-source-trello-clone-knban
- My Personal kanban
  http://greggigon.github.io/my-personal-kanban/

But neither of them are a complete tools, they do not provide WIP limits and the boards are not customizable.

As we can see, in the market exist plenty of powerful and complete Kanban tools, the wide majority are under commercial licenses. In the websites of these commercial Kanban tools we can observe that are being used by big companies. This means that Kanban should be a useful project management solution.

Apart from the commercial tools, there are few that can be used without any payment. This is a good way to use Kanban boards but all of them have limited features with the option to extend by paying.

Unfortunately, there is not any Open Source Kanban tool available through the browser to access the data stored on the cloud. At the moment there are available two Kanban tools prepared to install them on a private server. Moreover, both of them have a basic UI. There is also a desktop application that requires an installation on each device that has to be used.

In the past, there were other Open Source tools like **JAM Cicle** or **Simple-Kanban** but now they are not longer available.

It is also necessary to mention that there are some Scrum oriented Open Source tools that are not completely adapted to Kanban such as **Agilefant**, **ScrumDashboard**, **Planigle** or **Retrospectiva**.

3. THE TOOL

Figure 3.1.

KanBoard is the chosen name of the developed tool. KanBoard is a simple but powerful tool that covers the Kanban basics allowing users to visualize the entire project state in a nice and interactive board. It has been designed due to the fact that small teams collaborate in a unique interface, easily accessed through the web browser.

3.1 Purpose

The main purpose is to be a useful, clear and simple tool that could help the project management of the Agile software development teams. The goal consists in digitizing the physics Kanban boards into a browser application tool to improve the availability, the interaction and collaboration of a team. It pretends to allow the team to understand the workflow and the current status of the project.

From the beginning, the tool was designed to be understandable and user-friendly. Furthermore, it was intended to create something different of the current Kanban tools and at the same time that may be freely accessible to everybody.

The collaboration has been emphasized, as it is a very important need of the project team. Each member of the team has the chance to interact equally in the same project, adding, moving or editing tasks among the board.

KanBoard expects to become a helpful tool for students and small development teams with the purpose of improve their project management.

3.2 Features

In the following points are described the tool's most interesting features.

- **Unlimited number of Kanban boards available online –** KanBoard is a Kanban tool accessible through a web browser. All the information is stored in the server, so any user can access from different devices. Each user can create an unlimited number of Kanban boards.
- **Provides the current status of the entirely project –** With the main board view, the users can understand how the project is going on, for example by watching how much work they still have to do or how much work they have done.
- **Customizable Kanban boards with unlimited number of states –** Any user or team can create customizable boards creating as much states as they want. The states can be renamed to suit the needs of each user.
- **Use the mouse to drag & drop the items between states –** The items or tasks inside the board can be easily moved between the different states by simply dragging them.
- **Limit the items on a state to enforce the project workflow by WIP limits –** It is possible to limit the number of items inside a state. With the WIP limit, the project can adopt a better workflow avoiding unnecessary bottlenecks.
- **Items contained by specific information –** Each item or task is composed by a name, a description, a priority and an expected day to finish it. All the information of each task can be edited.
- **Edit, delete or share projects between a team –** The users can manage their projects by creating, editing or deleting them. Also they have the possibility to share a project with other members.
- **Real collaboration between members of a team in the same board with possibility to assign specific task between the members –** When a board is shared, the tasks can be assigned to any member of the team and each member can interact with the board at the same permission level.
- **Extremely easy to use, with no complications, user-friendly –** The application is characterized by implement a simple and easy to use UI. It has been specially designed to be the simplest and user-friendly as possible.
- **Completely free and Open Source –** All the source code is freely available on the Internet and the application is free to use, without any restriction or payment is needed.

Login and Register

At the beginning, when the user accesses for the first time, only has available a logging form in order to access the application. For those who are not registered there is a button to do it. The register form asks for a name, an email and a password.

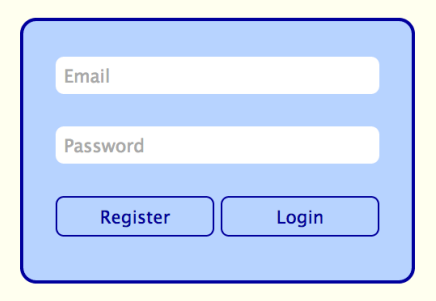


Figure 3.2.

Figure 3.3.

Internally, the system checks if exist other user registered with the same email, denying the register to those who are previously registered. Once the user is registered or logged, is redirected to the projects management page.

User Management

In the upper right corner of the screen appears the name of the user followed by the clickable icons for edit and logout. This small view remains visible while the user is logged in. Clicking to the logout icon, the user can close his session and he goes again to the initial login pag. The other icon, the pencil, opens a new view in where the personal information can be edited: the name and the email, the password or the option to delete the account.

Figure 3.4.

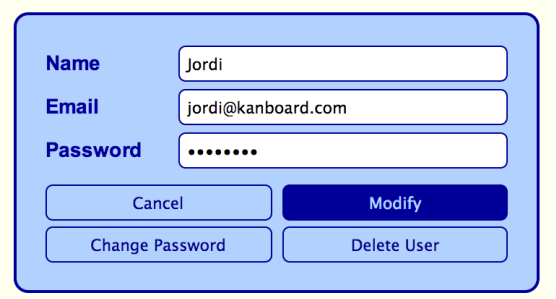


Figure 3.5.

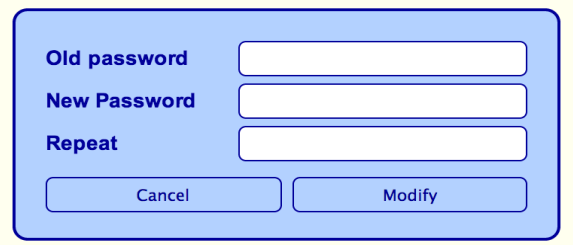


Figure 3.4.

Board Management

This is the first view that appears after the user has logged in. All the projects or boards of the user are listed. The board of each project is opened just by clicking on each task.

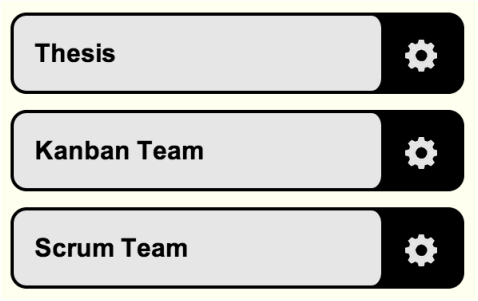


Figure 3.5.

There is also an option to create a new board.

Figure 3.6.

By pressing the settings icon in each specific board, a new options menu slides from the right. In this new menu, the user can see general information of the project, can edit the name of the project and can share or delete the project.

Figure 3.7.

The general information of the board shows when and who has created the board, when the task was modified and with whom it is shared. To share a project, the user has only to enter the email address of another member that has been already registered. The other member will notice in the projects page that she or he has a new project with a red border, giving him to know as a new member of that board.

Figure 3.8.

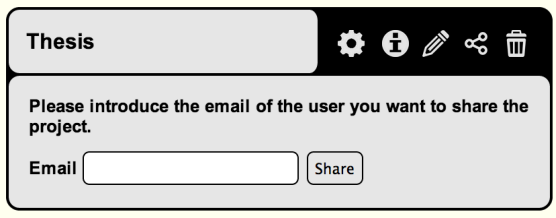


Figure 3.9.

If a user of a team wants to delete a shared board and he is not the owner or the creator, the board will disappear from his projects but the rest of the team will still be able to working on it. Furthermore, when someone deletes a board is asked to be sure about the action, because after that, all the information related or contained of that board is automatically deleted from the server.

The Kanban Board

This is the main view of the application, where contains all the items or tasks inside each state.

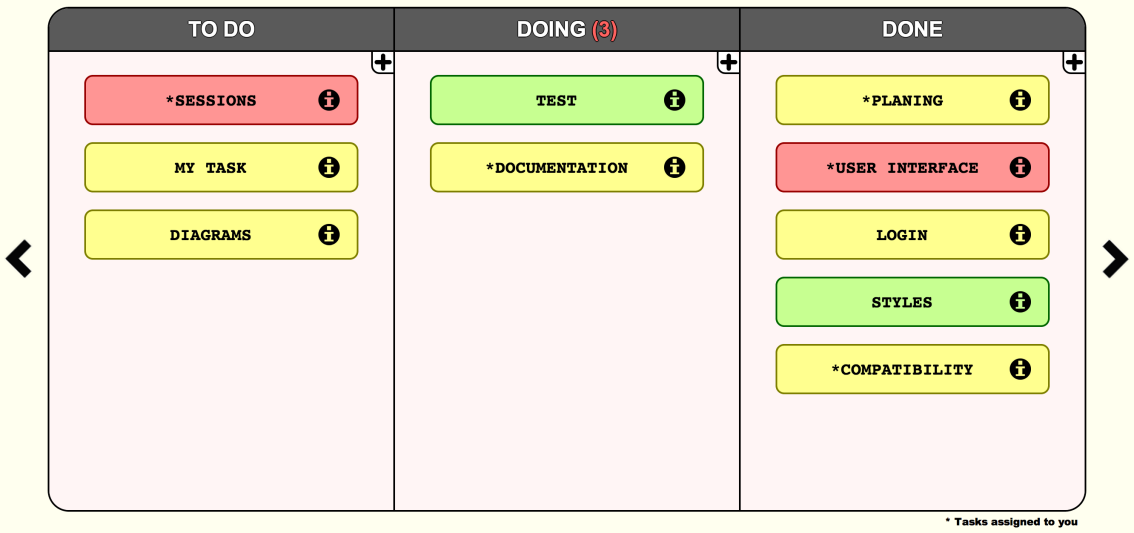


Figure 3.10.

The items can be easily moved around the different states by the drag and drop gesture, facilitating the interaction between the user and the system.

To create a new task, the user only has to click the plus icon under the state name and a new form will appear.

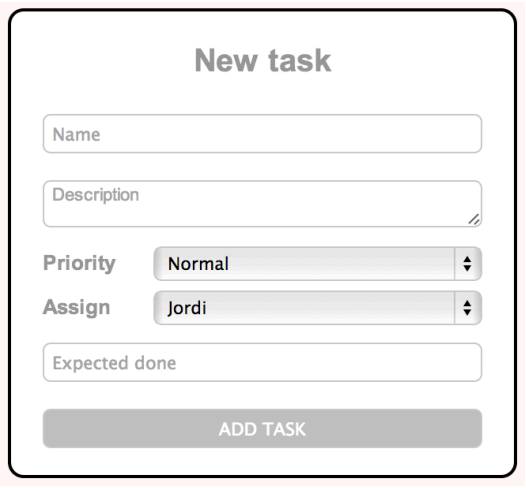


Figure 3.12.

Figure 3.11.

The attributes to fill in each task are the name, its description, the priority and the expected date to finish it. Even so, the system improves when the board is shared with other members. Each task can be assigned to any member of the team, sharing out the work between members as well as knowing which member is working in each project or who has already finished a specific task. If the board is shared, the logged user can see which tasks has assigned through a small star at the beginning of the task.

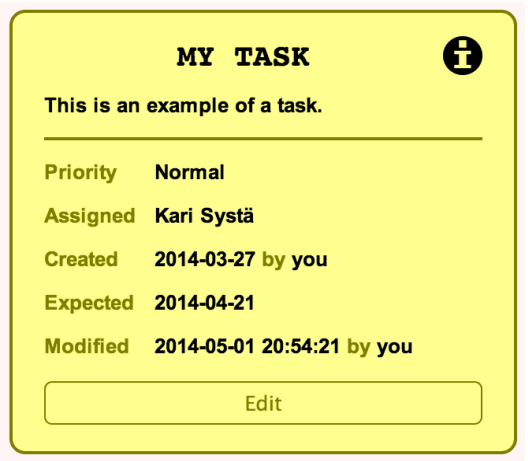


Figure 3.13.

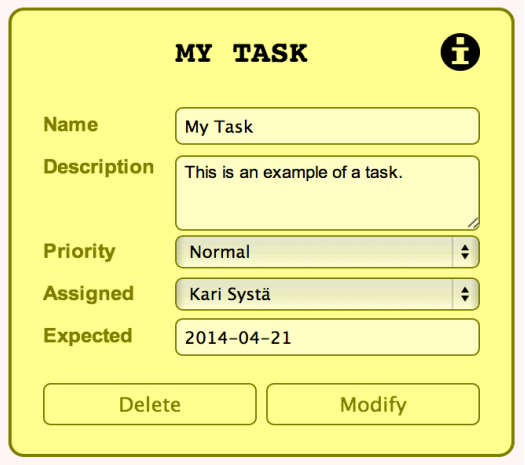


Figure 3.14.

To simplify the experience, when the user has to fill the "Expected done" field, appears a small calendar where he only has to choose the selected day.
In order to see in detail a specific task, the user can click in the information icon next to the name and a detailed view will be shown. Also there is the possibility to edit the related information, logging the last time modified.

By changing the priority of a task, the entirely item colour will change; green as a low priority, yellow as a normal one and red for a high priority.

When a board has more than three states, two arrows appear in the sides in order to scroll the board between the different states.

State Management

Clicking the top button “Edit states” in the board view, the user is redirected to a new page where the states of current board are listed in order. There is also a button to add a new state. To change the order of a specific state on the board, it just has to be swiped up or down till the desired place.

Figure 3.15.

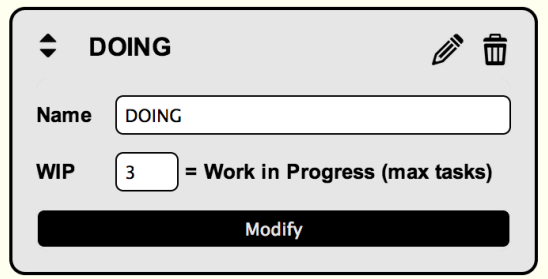


Figure 3.16.

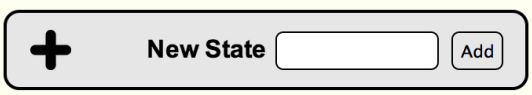


Figure 3.17.

Other interesting feature is the possibility to limit the WIP to get a better workflow. By clicking the pencil icon of edit, the user is able to change the state name limiting the items inside. After that, if a state has a limited WIP, in the main board that limit will be shown between brackets next to the name of the state. If the user wants to move an item to a state that has reached the WIP limit, an alert is shown denying the movement.

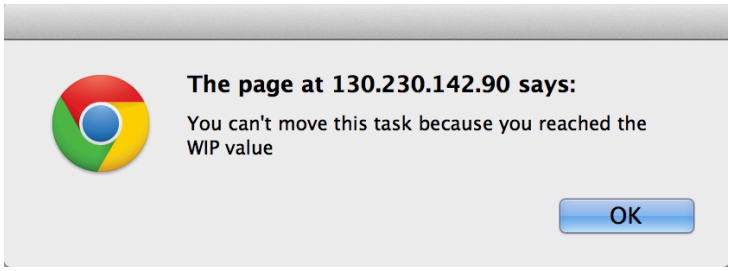


Figure 3.18.

3.3 Implementation

3.3.1 Requirements

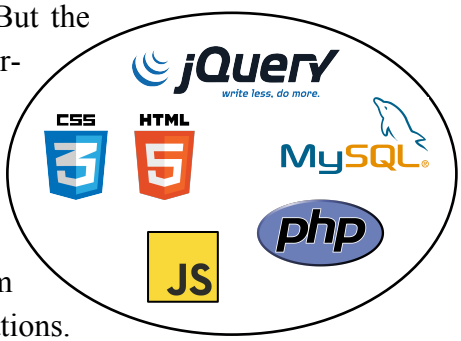
In order to properly perform the tool, some requirements were determined:

Requirement	*Priority*	*Functional*	*Non-functional*
The application has to be accessible online via browser.	Normal	-	Accessibility
The application has to be available through an IP address.	Normal	-	Availability
The application has to be designed at least for one browser.	Normal	-	Compatibility
The users need to authenticate themselves in order to access the application.	High	X	-
The private information from the users has to be stored encrypted.	Normal	-	Safety
The application has to prevent data-driven attacks.	Normal	-	Security
The system must show a visible board with the tasks inside.	High	X	-
The user can edit, share or delete their boards.	Normal	X	-
When a board is shared, all the members can interact with the same permission level.	Low	X	-
Only the creator can delete definitely the board.	Normal	X	-
The user has to be able to edit and order the different states of a board.	Normal	X	-

The system has to allow adding unlimited number of tasks or items.	High	X	-
An item has to be easily movable between states using the drag & drop gesture.	High	-	Usability
The system has to provide the user to limit the items in the states.	High	X	-
Each item has to be composed by a name, description, priority, and an expected finish date.	Normal	X	-
If a board is shared, the tasks or items can be assigned between the members.	Normal	X	-
The application has to be very intuitive and user-friendly.	High	-	Usability
The UI has to be simply and clear in order to facilitate the interaction between the user and the application.	Normal	-	Usability
All the source code has to be freely accessible under an Open Source license.	High	-	Open Source

3.3.2 Programing Languages

Given that it is a web-oriented application, the programing languages used are **HTML** for the structure and **CSS** for the styles. But the core or the main engine is written in **PHP**, a server-side scripting language designed for web development. Also there is a file with jQuery and **JavaScript** functions, where both are executed in the client side. jQuery allows to implement a better visual interaction between the user and the system allowing to select DOM elements and create animations. Finally, **SQL** queries are used to connect the MySQL database and the application.



3.3.3 Server Files

At present 16 files compose the application:

- 13 files with *php* extension, where are all the different views and connections to the database, composed by 1.076 code lines.
- 2 files with *css* extension, divided by the general styles and the board styles, composed by 588 code lines.
- 1 file with *js* extension, where are included all the Javascript and jQuery functions of the application, composed by 595 code lines.

There is also an images folder with all the icons in *png* extension.

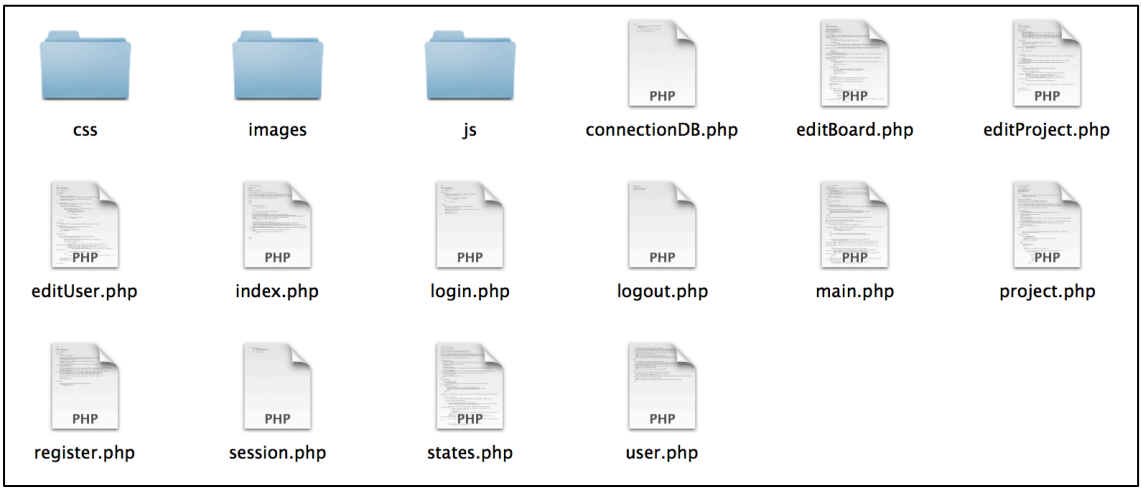


Figure 3.19. *View of all the files in the folder site.*

By combining all the files and the database script, there are in total 2.367 code lines.

3.3.4 Architecture

The application is based on a client-server architecture, where the server is waiting orders and the client is asking and receiving the data.

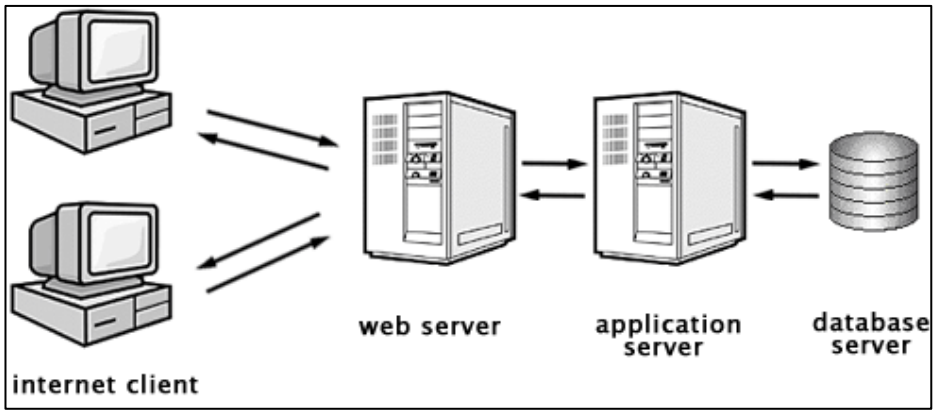


Figure 3.20. *Example of a client-server architecture [26]*

The application is composed by static and dynamic HTML. The static HTML is that part of code that is delivered to the user exactly as stored and displays the same information for all users. On the other hand, the dynamic HTML, or DHTML, is an umbrella term for a collection of technologies used together to create interactive and animated web sites by using a combination of a static markup language (such as HTML), a client-side scripting language (such as JavaScript), a presentation definition language (such as CSS), and the Document Object Model (DOM). DHTML allows scripting languages to change variables in a web page's definition language, which in turn affects the look and function of static HTML page content, after the page has been fully loaded and during the viewing process. Thus the dynamic characteristic of DHTML is the way it functions while a page is viewed, not in its ability to generate a unique page with each page load.

Another technology used is Ajax (Asynchronous JavaScript and XML), is composed by a group of techniques used on the client-side to create asynchronous web applications. With Ajax, the server can send asynchronously data (in the background) without interfering with the display and behaviour of the existing page. HTML and CSS can be used in combination to mark up and style information. The DOM is accessed with JavaScript to dynamically display, and allow the user to interact with the information presented. JavaScript and the XMLHttpRequest object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads.

DHTML is differentiated from Ajax by the fact that a DHTML page is still request/reload-based. With DHTML, there may not be any interaction between the client and server after the page is loaded; all processing happens in JavaScript on the client side. On the contrary, an Ajax page uses features of DHTML to initiate a request to the server to perform actions such as loading more content.

The application has only three full pages (index.php, project.php and states.php), where the browser has to reload each time are accessed, the rest of the contents are loaded but not showed due to the Ajax technique.

Currently the application is implemented on an Open LAMP stack platform, where the server machine has a Linux operating system, the web server is Apache HTTP Server, the database management system is MySQL and the web development language is PHP.

MVC

Since the beginning, it was intended to follow the MVC architectural pattern, trying to separate the data, the UI and the control logics.

- In the **View**, the user interacts with the system, the interface. Receives and shows the data from the model. e.g. of View files: index.php, main.php.
- The **Model** manages the system information, the database and the business model. e.g. of Model files: connectionDB.php, editUser.php.
- And the **Controller** replies the user events and involves changes in the model and the view parts. e.g. of the Controller files: login.php, project.php.

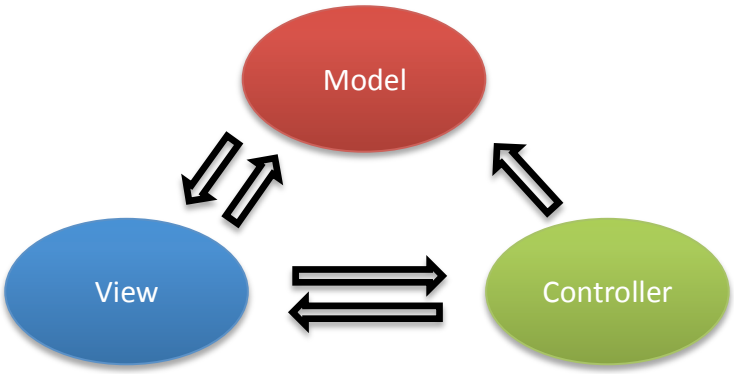


Figure 3.21. *MVC schema.*

3.3.5 Database

The schema of the database is used to store all the information related with the application is described below.

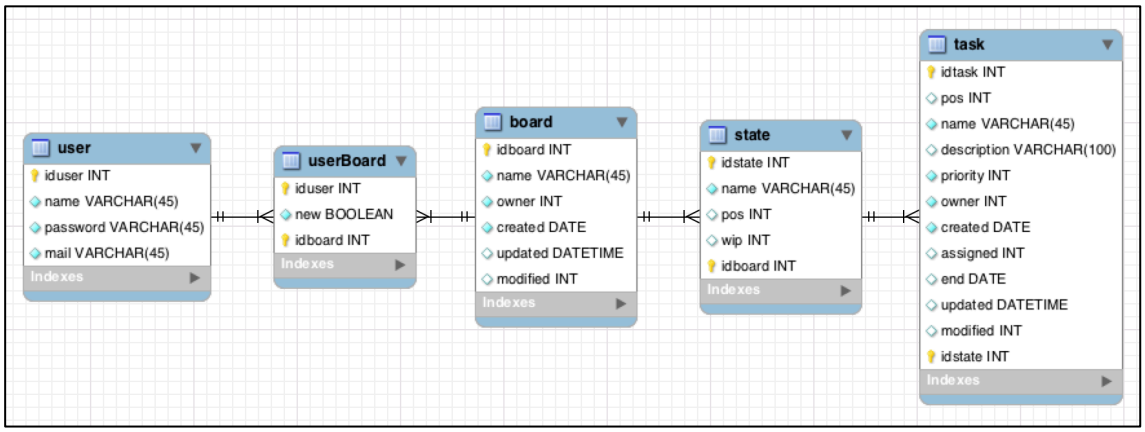


Figure 3.22. *Schema of the database.*

The *user* table has four attributes: a unique ID, a name, a password and an email. All of them cannot be NULL. Each user can own *n* number of boards, and each board can be owned by *n* number of users, for that reason the table *userBoard* can be useful. The *userBoard* table link the users with their boards and each board will be linked to all members who share. The *new* Boolean attribute is used to notify a user when he is added to a shared board. A part from that, each board has an ID, a name, an owner, the date when it was created and updated and also who modified it for the last time. The attributes owner and modified refer to a specific user, for those reasons are Integer types in order to link with the ID of the user. Each board has *n* number of states and each state contains *n* number of tasks. The attributes of the *state* table are composed by an ID, a name, a position, a WIP value and the ID of the board that belongs. The position attribute is very useful to show and sort the states in the board. And finally the *task* table contains twelve attributes. Among them, there are an ID, a position, a name, a description, the priority, the owner or creator of the task, the date when it was created, the user assigned (if it is a shared board), the expected finished date, when and who modified for the last time and the ID of the state where is located.

3.3.6 Development

In all the coding stage, a very simple physical board has been used to help the development process. It was made by black tape in a wall, filled by small written cards.

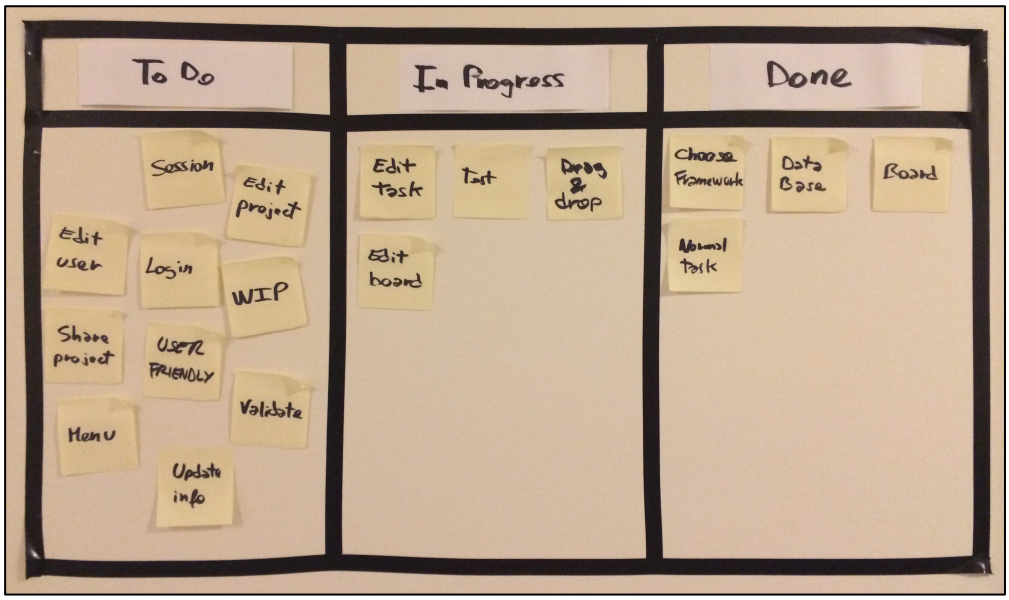


Figure 3.25. *Board used during the development.*

The decision of which framework or programing language should be used was the first task. Evaluate the best option between PHP, Java or Ruby took some days. Finally, the PHP language was chosen together with JavaScript.

First of all some sketches were defined to have an initial idea of how the main table would look like. At the beginning the idea was to represent the tasks in square shapes, having two columns in each state, but finally it has been decided to represent elongated shapes only with the task name. What was clearly defined was that the main view will be a big board with three states.

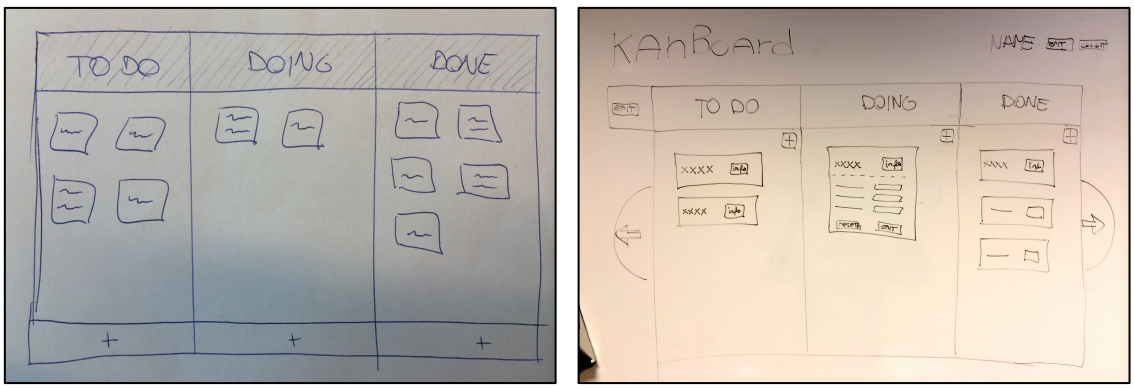


Figure 3.23. *Firsts sketches of the main board view.*

In the first database schema the users where directly linked with the tasks, and then, the tasks were inside a state. Later, with the inclusion of multiple tables and the sharing option, the schema was changed to the actual.

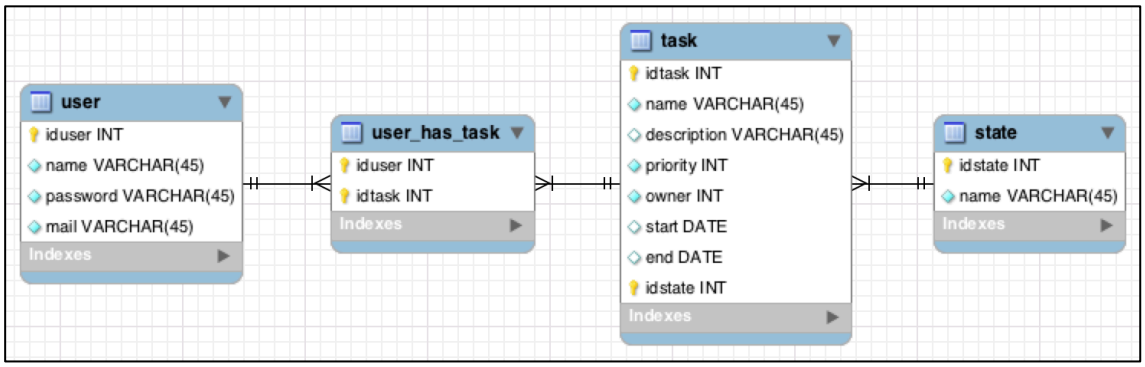


Figure 3.24. *First database schema*

The first PHP file created was the connection to the database, composed by a simple function that links with the specific local database.

```
<?php
	$con=mysql_connect("localhost", "root", "kanban");
	if (!$con){
		die('Could not connect: ' . mysql_error());
	}
	mysql_select_db("kanban_DB", $con);
?>
```

Programme 3.1. *Connection to the database.*

After that, the connection file was included in another file where a simple HTML table was created. The table was modified with some PHP instructions, allowing modifying the size according to the database elements. By introducing manually some elements in the database was enough to see how the board looked with some states and tasks inside.

TO DO	DOING	DONE
TEST	TEST2 TEST3	

Figure 3.25.

The next step was the implementation of adding tasks and states directly in the board. To perform this, a form was showed when clicking a button. Once the form was fullfilled, the information submitted remained stored in the server.

Then, an information button was added next to the task name. By pressing this button, all the information of the task was showed below, moving down the other tasks of that state. Once the information of a task was accessible, a button was added to allow editing it. Also, the different colour for each task was implemented depending on their priority.

After that, some research about how to move the tasks by the drag & drop gesture was done. The best way found was a jQuery UI component called *sortable*.

```
$(document).ready(function() {
	$('.laneClass').sortable({
		connectWith: '.laneClass',
		stop: function (event, ui){
			var order = $(this).sortable("serialize");
			$.post("editBoard.php", order, function(){});
			var idtask = ui.item.attr('id');
			var idstate = ui.item.parent().attr('id');
			$.post("editBoard.php",
			{
				idtask:idtask.split('task_')[1],
				idstate:idstate.split('state_')[1],
				moveTaskLane:"ok"
			},
			function(){
			});
		}
	}).disableSelection();
});
```

Programme 3.2. *jQuery sortable function adapted to the application needs.*

With the previous function, the items inside the *laneClass* class are able to move around the board by drag & drop. The stop event allows running actions when the user has dropped the item, sending by post and saving the ID of the task with the ID of the destination state.

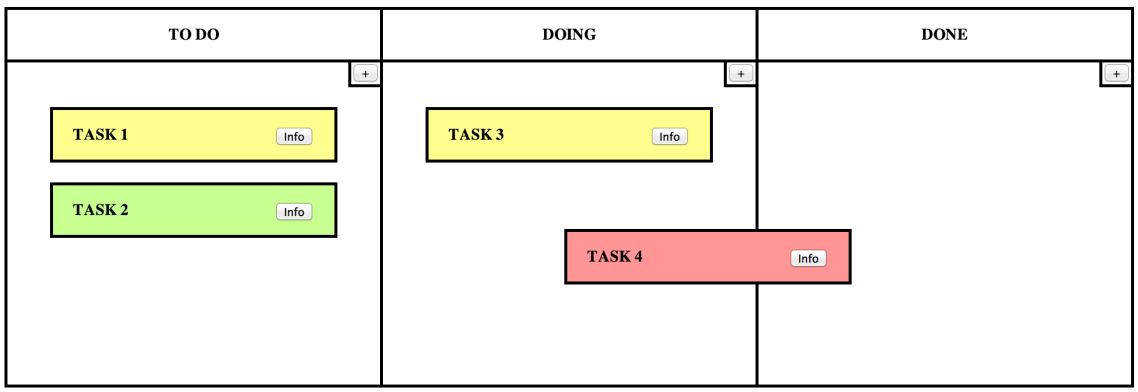


Figure 3.26.

Over this time, the users were not yet linked to the tasks. The following step was the edition of the states. In another view, the states were listed in vertical with the options to edit their name or delete them. After that, there were provided also with drag & drop in order to sort them in the board, but in this case, limited to a vertical movement.

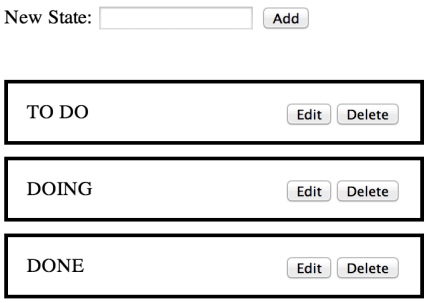


Figure 3.30.

The following step was the implementation of the login and consequently, the implementation of the sessions.

```
$query=mysql_query("SELECT * FROM user WHERE mail='$mail'");
$row=mysql_fetch_array($query);

$logged=array('iduser'=>$row['iduser'],'name'=>$row['name'],'mail'=>$mail);
$_SESSION['logged']=$logged;
```

Programme 3.3. *Part of the login file.*

The previous code belongs to a part of the login.php file, where the *iduser*, *name*, and *mail* attributes are saved in the web page session.

From that moment, the login form was compulsory to access to the application. What is more, a form that allows registering the new users was created. The only restriction in the register process was that a new user could not have the same email address than another that has been already registered.

Figure 3.31.

Figure 3.32.

In that moment the database was not well designed, so, the database design was changed to fit with the new needs. When a user was able to login into the system, it was needed a unique boards for each one. Because of that, a new view was created in order to provide the different bards of each user. Each board was endowed with the possibility to edit its name or delete it entirely.

Figure 3.33.

At the same time was developed a simple menu in the top that allows the interaction between the different views.

Then, to be able to move between the different states of the board, the lateral displacement was improved with two arrows on the sides.

Next, were implemented the WIP limits, where the user can limit a state with a number, then, it is visible on the board and it is not allow to move task in a state that has reached the limit.

Figure 3.34.

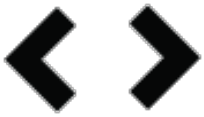


Figure 3.35.

After some exhaustive test and when everything was working fine, the sharing board feature was implemented. When someone shares a board with an other user by the email address, the database link the board with the new user notifying it with a different border colour that disappears when the user interact with the board.

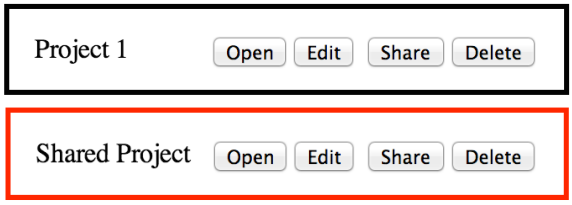


Figure 3.36.

Another feature implemented was the last modified log, saving the last time that a board or a task is modified, either a state change or editing the information of a task. In the shared boards, the responsible of the last update is saved and showed in the board

information. Each project is provided with a general information about the entirely board, such as when and who created and modified it for the last time and which are the members who are sharing that board. The options of name edition, sharing and deleting the board were added.

Figure 3.37.

The basic feature to allow the user edit to his information was also implemented. Different forms were designed in order to be able to change the name, the mail or the password. And the possibility to delete the user from the system was implemented as well. When a user unsubscribes the application all his boards and tasks are deleted to prevent waste space in the database.

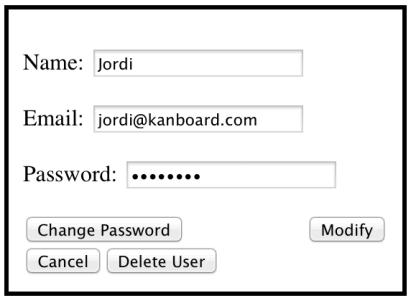


Figure 3.38.

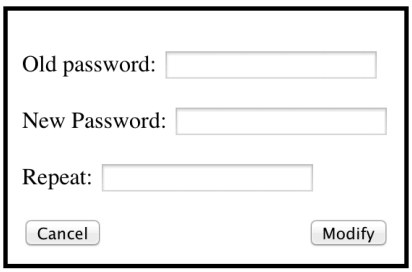


Figure 3.39.

After everything was almost fully implemented, different situations that could destabilize the system were tested, and all the forms were carefully validated.

Figure 3.40.

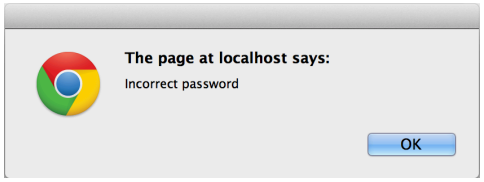


Figure 3.41.

The last part of the coding was making it user-friendly with a nice GUI. In order to improve the user experience and helped by jQuery functions, some movements were converted to animations. The CSS styles were modified in detail to get a better good-looking and some buttons were replaced by intuitive icons. Different background colours and clear views were added. For the new registered users, a starter board with a single example task was created automatically to a better understanding of the tool.

The next picture shows how looks the finished tool in the board view:

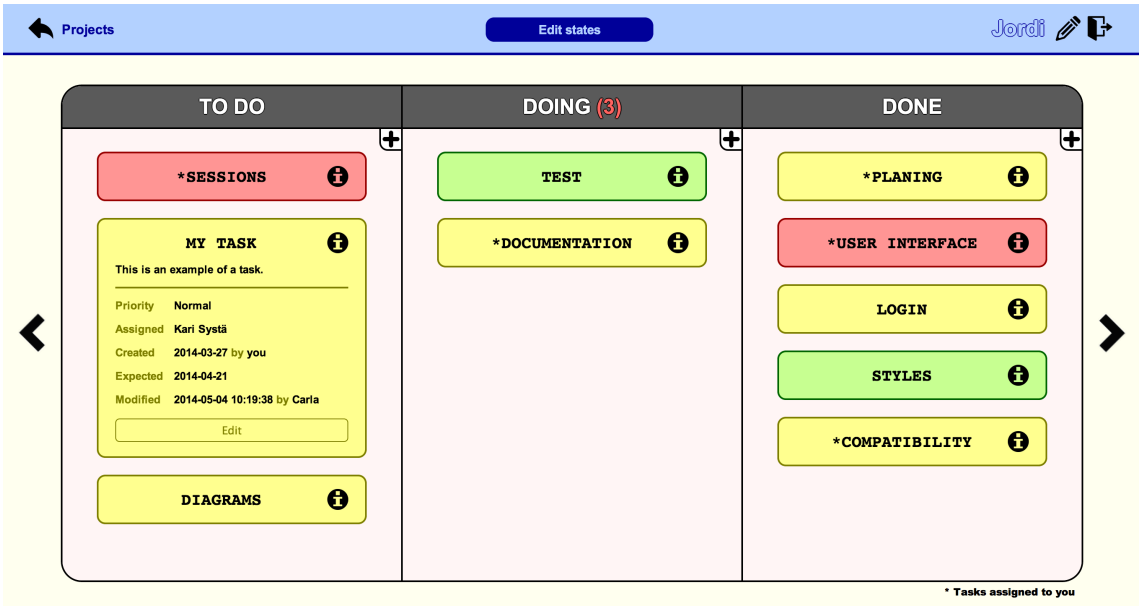


Figure 3.42. *Board view of the application.*

3.3.7 Testing

The application has been tested by some users using the **Exploratory Testing** technique. During the development process, the modules were individually tested as well as each time they were integrated. Once the application was ready, the IP address where the application is located, was given to few users in order to they could try, test and report errors. After that the majority of errors were resolved, the application was tested by a group of experts, giving their feedbacks for future improvements.

3.3.8 Configuration Management

During all the development process, the configuration management tool used was GitHub. As KanBoard is an Open Source tool, the GitHub repository allows the interested users to view or download the code, even they can fork or clone it and continue working in parallel.

GitHub has been used as backup; committing the code each time there was a stable module. The commits were improvements of previous versions or the integration of new features. Because of that, there is available a complete history of the improvements of the project, being able to see the changes in each commit.

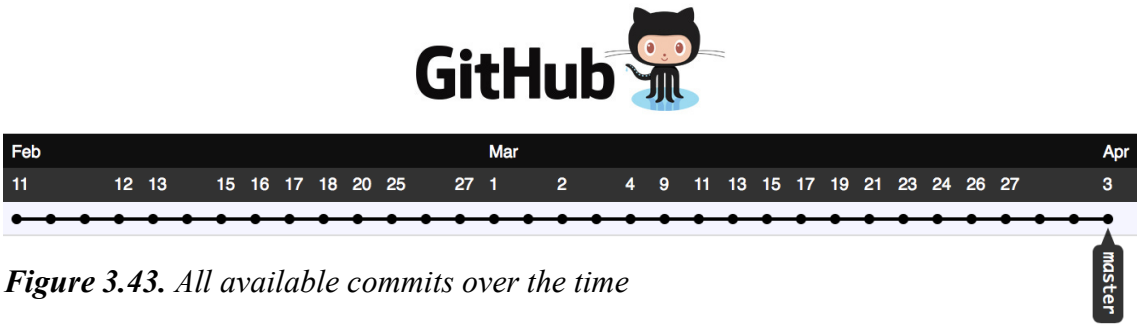


Figure 3.43. *All available commits over the time*

The repository is open and available to everyone in the link below:
https://github.com/jordimd/kanbanThesis

3.4 Evaluation

At this point, the tool can be called as an Alfa version, where all the main features are achieved. It can be improved in many aspects but it works in what proclaims.

Even so, some professionals have tested the tool and they made a lot of constructive reviews. Here there are few of them:

I think the tool is nice and simple in general. However, there exists a plethora of similar tools already, e.g. Agile Zen. The tool seems to work quite well in Chrome. I would have liked more if the tool would have had more responsive UI design. Now, if I open the browser on small screen (e.g. laptop) and move the browser window to larger screen, the board size does not scale automatically, but I have to go back to project selection and then come back.
Another issue that caught my eye was that the name of the item is rather short. For example I tried to create a ticket "Implement feature X" but the field was too short for that. I would prefer longer name field.
Also buttons in the ticket (modify and edit) look a little bit like they were disabled and you cannot press them. Light gray color might not be a best option here.
One more thing, you can select the expected completion date from the past. I am not sure if this is intentionally.

Anyway, the tool looks simple and easy to use, so it might be good choice for students in project work course.

Wellu.

Great work!
Overall, nice, simple design, also nice "animations", so what I'm commenting are just minor improvement suggestions.

Add tooltips to icons, if possible.
Consider enabling sending a registration invitation email to new project members who has not registered yet.
If I click pen icon on top left besides my user id, I get a view with buttons cancel, modify, change password & delete user. Modify goes back to "normal view" which was surprising, since I expected I would be somehow modifying the account
It took me a while to find out how to add new tasks etc. (that is, to find out that the project name is a clickable item. Consider adding some clue about that.
Tasks part is done wonderfully! Grab-drag was a nice feature, and I guessed intuitively that one immediately.
Modifying stages: warn the user when closing e.g. to do -field, the changes made disappear if not saved (not pressed "modify" -button), well, not much re work if this mishap happens, though.

Jarmo.

The tool looks quite nice and I like the fact that the UI is not cluttered with features. I see it could be used in project course or SW engineering methodologies course.

However, to improve the UX, at least in my opinion :) Some of these are really nitpicking, but I thought any feedback could be valuable.
First, you should seriously add support for Firefox. I don't have Chrome installed on my work PC.
Cancel buttons would more natural on the right side. Especially in personal details, the cancel button gets lost.
Please, add some tool tips to the icons. It is frustrating to guess the meaning of the buttons: "What does a giant plus sign do?"
Pop up for adding people to the project appears over the browser toolbar. And it actually says that the person is "now in the project". I would expect that there would be some kind of acceptance by the added person. What happens if I delete a project where I have added other people?
It took a while to understand that the project name is a clickable object to edit the actual content. A tool tip and highlighting with mouseover would help. It was also unintuitive to have the gear symbol to open and close the tools.

[...]

There was my shot at this. Keep up the good work!

Marko.

As it clearly can be seen, the application needs to be improved in some details. Maybe one of the biggest issues could be the compatibility with different browsers. It is true that it is one of the headaches of web developers, but after all, is a matter time. It was preferred to cover more features instead to make the compatibility. That is why it was limited to run only under Google Chrome, to avoid significant problems in other platforms. Another issue related with browsers was that, even running the Google Chrome on Windows and Mac OS X, some information is showed in different way. Other aspects such as tips or a little help about the features could be a great idea to implement. The buttons that allow moving the table trough the different states should only be visible when there are more states on this side. And also the chosen colours or few buttons aspect can be modified. Some of these points should be checked, as soon as possible, in order to improve the user experience.

Even so, the tool is distinguished by the simplicity and the nice user interface. While other tools try to cover as much features they can, KanBoard offer a Kanban solution without complications neither restrictions. Surely, now is the unique Open Source Kanban tool with the previous features available for free on Internet.

As future improvements, there are a lot of new features that could be implemented. For example the possibility to decide if the board has a backlog state by default, showing it as none main view of the board, maybe in a clickable left side. Or the possibility to choose some state templates at the time to create a new board, like basic board: [TO DO, DOING, DONE], software development: [TO DO, DEV, TEST, RELEASE, DONE] or [NEXT, ANALYSIS, DEV, ACCEPT, PROD]. Another feature to implement could be the monitoring and analysing the data, allowing the visualization of different charts like Cumulative flow, Cycle time, Task distribution or Block resolution time.

If the project continues in the future, KanBoard could become a reference as an Open Source Kanban tool.

4. CONCLUSIONS

Agile software development is a group of methods based on iterative and incremental development. These methods were emerged, from the necessity of find new alternatives to the exist ones. Nowadays the most popular and used methods are Scrum and Scrum variants. Even so, Kanban is proving useful to teams doing Agile software development but equally it is gaining traction with teams taking a more traditional approach. Kanban is being introduced as part of Lean initiative to morph the culture of organizations and encourage continuous improvement.

Kanban seems like such a small change and yet it changes everything about a business and approach to change management. Thus, Kanban is based on the general idea of all the Agile methods within the improvement of the team work. Kanban is an approach to introducing change to an existing software development lifecycle or project management methodology.

Work In Progress (WIP) limit is the most representative Kanban feature. Using WIP, allow optimizing the Kanban system to improve the workflow, collect metrics to analyse flow, and even get leading indicators of future problem. WIP should be limited and something new should be started only when an existing piece of work is delivered or pulled by a downstream function. Because WIP is limited in a Kanban system, anything that becomes blocked for any reason tends to clog up the system. If enough work items become blocked the whole process stops. This has the effect of focusing the whole team and the wider organization on solving the problem, unblocking the item and restoring the flow. The effect of limiting WIP provides predictability of cycle time and makes deliverables more reliable.

Nowadays there are used plenty of Kanban tools in the market but most of them are commercial license. However, there are a couple complete Open Source Kanban tools available to host on own servers, none of them are cloud based. Considering, therefore, the importance of current demand by software development groups and the fact to study Kanban closer, an Open Source Kanban tool was created to cover these needs.

Finally, is expected to become a helpful tool for many students, even small development teams with the purpose of improve their project management.

REFERENCES

[1] Agile Manifesto for Agile Software Development [www]. [accessed on 18.12.2013]. Available at: http://agilemanifesto.org

[2] Examining the Agile Manifesto [www]. [accessed on 20.5.2014]. Available at: http://www.ambysoft.com/essays/agileManifesto.html

[3] Writing the Agile Manifesto [www]. [accessed on 15.5.2014]. Available at: http://martinfowler.com/articles/agileStory.html

[4] 8th Annual State of Agile Survey. VerisonOne Inc. 2014. 17 p. Available at: http://stateofagile.versionone.com

[5] Pronciples of Lean [www]. [accessed on 12.5.2014]. Available at: http://www.lean.org/WhatsLean/Principles.cfm

[6] Lean Software Development [www]. [accessed on 12.5.2014]. Available at: http://msdn.microsoft.com/en-us/library/hh533841.aspx

[7] Scrum Methodology and Project Management [www]. [accessed on 10.1.2014]. Available at: http://www.mountaingoatsoftware.com/agile/scrum

[8] Henrik Kniberg, Mattias Skarin. Kanban and Scrum – making the most of both. InfoQ 2010, C4Media Inc. 102 p.

[9] Andrea Tomasini, Martin Kearns. InfoQ. agile42, 2012. 35 p.

[10] What is Extreme Programming? [www]. [accessed on 23.1.2014]. Available at: http://xprogramming.com/xpmag/whatisxp

[11] What is Kanban Agile [www]. [accessed on 20.1.2014]. Available at: http://www.versionone.com/what-is-kanban/

[12] What is Kanban? [www]. [accessed on 21.1.2014]. Available at: http://kanbanblog.com/explained/

[13] Kanban – an alternative path to agility [www]. [accessed on 18.2.2014]. Available at: http://www.djaa.com/kanban-alternative-path-agility

[14] Agile/Scrum, Kanban & Waterfall [www]. [accessed on 3.2.2014]. Available at: https://sprintly.uservoice.com/knowledgebase/articles/98887-agile-scrum-kanban-waterfall

[15] Kanban vs Scrum Myths and Hype [www]. [accessed on 8.2.2014]. Available at: http://blogs.versionone.com/agile_management/2012/01/06/1-kanban-vs-scrum-myths-hype/

[16] How we chose our Kanban Tool [www]. [accessed on 19.12.2013]. Available at: http://www.multunus.com/blog/2013/03/how-we-chose-our-kanban-tool/

[17] Kanban for Lean Project Management with agilezen.com [www]. [accessed on 28.3.2014]. Available at: http://www.agileweboperations.com/kanban-for-lean-project-management-with-agilezencom

[18] Henrik Kniberg. Scrum and the XP from the Trenches. InfoQ 2010, C4Media Inc. 128 p.

[19] Top Agile Open Source Tools [www]. [accessed on 28.3.2014]. Available at: http://tools.projekt-log.de

[20] Agile Tools & Reviews [www]. [accessed on 4.4.2014]. Available at: http://agiletools.info

[21] Agile Kanban Tools [www]. [accessed on 2.4.2014]. Available at: http://agilelion.com/agile-kanban-library/agile-kanban-tools

[22] Michael Sahota. An Agile Adoption and Transformation Survival Guide: Working with Organizational Culture. 2012, Michael Sahota. 67 p.

[23] 7 Kanban Board Tools for Project Collaboration [www]. [accessed on 8.4.2014]. Available at: http://collaboration.about.com/od/projectmanagement/tp/7-Kanban-Board-Tools-For-Project-Collaboration.htm

[24] jQuery UI – Droppable [www]. [accessed on 29.1.2014]. Available at: http://jqueryui.com/droppable/

[25] jQuery UI – Sortable [www]. [accessed on 29.1.2014]. Available at: http://jqueryui.com/sortable/

[26] Client-server technology [www]. [accessed on 18.5.2014]. Available at: http://www.dbrange.com/technology.html

[27] Jesper Boeg. Priming Kanban. 2nd Edition February 2012, Trifork A/S. 83 p.

[28] What is Exploratory Testing? [www]. [accessed on 15.5.2014]. Available at: http://www.satisfice.com/articles/what_is_et.shtml

[29] Non-Functional Requirements [www]. [accessed on 18.5.2014]. Available at: http://www.methodsandtools.com/archive/archive.php?id=113

[30] Scrum Guide [www]. [accessed on 28.5.2014]. Available at: https://www.scrum.org/Scrum-Guide

[31] The Principles of Lean Software Development [www]. [accessed on 29.5.2014]. Available at: https://www.ibm.com/developerworks/community/blogs/ambler/entry/principles_lean_software_development?lang=en

[32] Agile Software Development Method [www]. [accessed on 22.4.2014]. Available at: https://sea.ucar.edu/best-practices/agile

[33] ApacheMySQLPHP – Community Help Wiki [www]. [accessed on 30.5.2014]. Available at: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu

[34] How To Install Linux, Apache, MySQL, PHP (LAMP) Stack on Ubuntu [www]. [accessed on 30.5.2014]. Available at: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu

[35] GitHub Repository – kanbanThesis [www]. Available at: https://github.com/jordimd/kanbanThesis

APPENDIX

Installation Process

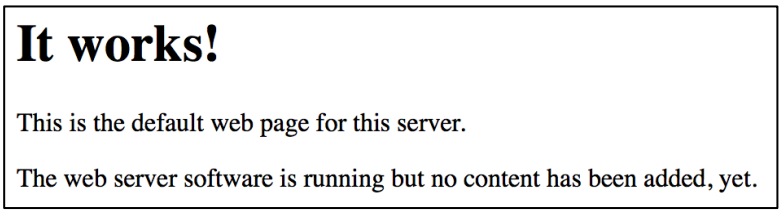
This is a small tutorial on how to install the application in any Linux OS host. The installation process will be the same for any Linux distribution; in this case the server has installed an **Ubuntu 12.04.4 LTS (Precise Pangolin)** OS.

In order to work, we need to access the server by the terminal in the same machine or maybe remotely helped by the SSH protocol.

First of all we should update the system, after that we need to install the Apache webserver. Apache can be installed by inserting the following command:

```
$ sudo apt-get install apache2
```

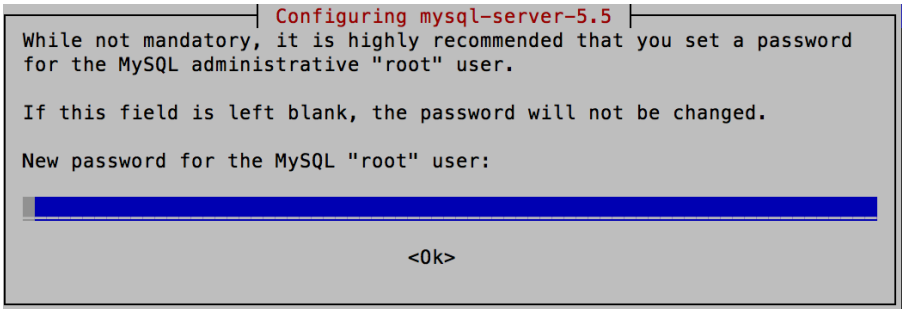
Currently is installed the **Apache/2.2.22 (Ubuntu)** version. To check if Apache is installed properly we can introduce the server IP address in the browser. We should obtain a page with the words “It works!”



The next to install is the MySQL database management system. In the terminal we have to type:

```
$ sudo apt-get install mysql-server libapache2-mod-auth-mysql
```

During the process the system will ask us to create a new password to access the database, where we should introduce a new one.



Now the application is running the MySQL **Ver 14.14 Distrib 5.5.35**.

The next step is installing the PHP. PHP is a web scripting language that is widely used to build dynamic webpages.

To install PHP, we should type in the terminal this command:

```
$ sudo apt-get install libapache2-mod-php5 php5-mysql
```

The server has installed the **PHP 5.3.10-1ubuntu3.9** version.

It may also be useful to add PHP to the directory index, to serve the relevant PHP index files:

```
$ sudo nano /etc/apache2/mods-enabled/dir.conf
```

Add *index.php* to the beginning of index files. The page should now look like this:

```
<IfModule mod_dir.c>
   DirectoryIndex index.php index.html index.cgi index.pl in-
   dex.xhtml index.htm
</IfModule>
```

In order to check if the entire LAMP stack has been correctly installed, we can check creating a PHP file in the site folder (usually /var/www/) called info.php. The file has to content only the following three lines:

```
<?php
phpinfo();
?>
```

After that, we should restart the Apache so that all of the changes take effect:

```
$ sudo service apache2 restart
```

And then by visiting the created file through the browser, we should observe all the stack details.

Once we have the running the server, we should upload the files in the site directory. If we are working remotely, we should install first the FTP to transfer the files to the server host.

First we should take ownership of the web root:

```
$ sudo chown -R pi /var/www
```

Next, install vsftpd:

```
$ sudo apt-get install vsftpd
```

After, we need to make changes in the vsftpd.conf file:

```
$ sudo nano /etc/vsftpd.conf
```

And change *anonymous_enable=YES* to *anonymous_enable=NO* and uncomment *local_enable=YES* and *write_enable=YES*

Now we will be able to upload remotely all the files to the server.

The last step is the creating the database, it can be done by entering commands in the terminal or using tools to handle the administration through the web browser such as **phpMyAdmin**. After the database is created, the application is completely installed.

All the needed files and the database script can be found on the GitHub repository:

https://github.com/jordimd/kanbanThesis

PHP Version 5.3.10-1ubuntu3.9

System	Linux kanban 3.2.0-38-virtual #61-Ubuntu SMP Tue Feb 19 12:37:47 UTC 2013 x86_64
Build Date	Dec 12 2013 04:10:01
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d
Additional .ini files parsed	/etc/php5/apache2/conf.d/mcrypt.ini, /etc/php5/apache2/conf.d/mysql.ini, /etc/php5/apache2/conf.d/mysqli.ini, /etc/php5/apache2/conf.d/pdo.ini, /etc/php5/apache2/conf.d/pdo_mysql.ini
PHP API	20090626
PHP Extension	20090626
Zend Extension	220090626
Zend Extension Build	API220090626,NTS
PHP Extension Build	API20090626,NTS
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, tls
Registered Stream Filters	zlib.*, bzip2.*, convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, mcrypt.*, mdecrypt.*

apache2handler

Apache Version	Apache/2.2.22 (Ubuntu)
Apache API Version	20051115
Server Administrator	webmaster@localhost
Hostname:Port	127.0.0.1:80
User/Group	www-data(33)/33
Max Requests	Per Child: 0 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 5
Virtual Server	Yes
Server Root	/etc/apache2
Loaded Modules	core mod_log_config mod_logio prefork http_core mod_so mod_alias mod_auth_basic mod_authn_file mod_authz_default mod_authz_groupfile mod_authz_host mod_authz_user mod_autoindex mod_cgi mod_deflate mod_dir mod_env mod_mime mod_negotiation mod_php5 mod_reqtimeout mod_setenvif mod_status

Apache Environment

Variable	Value
HTTP_HOST	130.230.142.102
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_COOKIE	PHPSESSID=ul4md1307ea7t3m0hsl6jopch7
HTTP_USER_AGENT	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_3) AppleWebKit/537.76.4 (KHTML, like Gecko) Version/7.0.4 Safari/537.76.4
HTTP_ACCEPT_LANGUAGE	en-us
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_CONNECTION	keep-alive
PATH	/usr/local/bin:/usr/bin:/bin
SERVER_SIGNATURE	<address>Apache/2.2.22 (Ubuntu) Server at 130.230.142.102 Port 80</address>
SERVER_SOFTWARE	Apache/2.2.22 (Ubuntu)
SERVER_NAME	130.230.142.102
SERVER_ADDR	10.1.1.20
SERVER_PORT	80
REMOTE_ADDR	83.47.110.70
DOCUMENT_ROOT	/home/ubuntu/www
SERVER_ADMIN	webmaster@localhost
SCRIPT_FILENAME	/home/ubuntu/www/info.php
REMOTE_PORT	54834
GATEWAY_INTERFACE	CGI/1.1
SERVER_PROTOCOL	HTTP/1.1
REQUEST_METHOD	GET
QUERY_STRING	*no value*
REQUEST_URI	/info.php
SCRIPT_NAME	/info.php

HTTP Headers Information

HTTP Request Headers	
HTTP Request	GET /info.php HTTP/1.1
Host	130.230.142.102
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Cookie	PHPSESSID=ul4md1307ea7t3m0hsl6jopch7
User-Agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_3) AppleWebKit/537.76.4 (KHTML, like Gecko) Version/7.0.4 Safari/537.76.4
Accept-Language	en-us
Accept-Encoding	gzip, deflate
Connection	keep-alive
HTTP Response Headers	
X-Powered-By	PHP/5.3.10-1ubuntu3.9
Vary	Accept-Encoding
Content-Encoding	gzip

PHP Variables

Variable	Value
_COOKIE["PHPSESSID"]	ul4md1307ea7t3m0hsl6jopch7
_SERVER["HTTP_HOST"]	130.230.142.102
_SERVER["HTTP_ACCEPT"]	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
_SERVER["HTTP_COOKIE"]	PHPSESSID=ul4md1307ea7t3m0hsl6jopch7
_SERVER["HTTP_USER_AGENT"]	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_3) AppleWebKit/537.76.4 (KHTML, like Gecko) Version/7.0.4 Safari/537.76.4
_SERVER["HTTP_ACCEPT_LANGUAGE"]	en-us
_SERVER["HTTP_ACCEPT_ENCODING"]	gzip, deflate
_SERVER["HTTP_CONNECTION"]	keep-alive
_SERVER["PATH"]	/usr/local/bin:/usr/bin:/bin
_SERVER["SERVER_SIGNATURE"]	<address>Apache/2.2.22 (Ubuntu) Server at 130.230.142.102 Port 80</address>
_SERVER["SERVER_SOFTWARE"]	Apache/2.2.22 (Ubuntu)
_SERVER["SERVER_NAME"]	130.230.142.102
_SERVER["SERVER_ADDR"]	10.1.1.20
_SERVER["SERVER_PORT"]	80
_SERVER["REMOTE_ADDR"]	83.47.110.70
_SERVER["DOCUMENT_ROOT"]	/home/ubuntu/www
_SERVER["SERVER_ADMIN"]	webmaster@localhost
_SERVER["SCRIPT_FILENAME"]	/home/ubuntu/www/info.php
_SERVER["REMOTE_PORT"]	54834
_SERVER["GATEWAY_INTERFACE"]	CGI/1.1
_SERVER["SERVER_PROTOCOL"]	HTTP/1.1
_SERVER["REQUEST_METHOD"]	GET
_SERVER["QUERY_STRING"]	*no value*
_SERVER["REQUEST_URI"]	/info.php
_SERVER["SCRIPT_NAME"]	/info.php
_SERVER["PHP_SELF"]	/info.php
_SERVER["REQUEST_TIME"]	1402439678